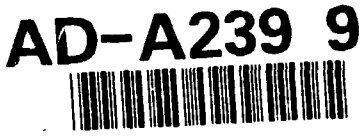

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

NUMERICAL INVESTIGATION OF THE
EFFECT OF LEADING EDGE GEOMETRY
ON DYNAMIC STALL OF AIRFOILS

by

Lt. Col. Steven P. Grohsmeyer
September 1990

Thesis Advisor: Dr. J. A. Ekaterinaris
Thesis Co-Advisor: Dr. M. F. Platzer

Approved for public release; distribution is unlimited

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1a REPORT SECURITY CLASSIFICATION: Unclassified	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION: Naval Postgraduate School; 6b OFFICE SYMBOL (If applicable): 31	7a NAME OF MONITORING ORGANIZATION: Naval Postgraduate School
6c ADDRESS (City, State, and ZIP Code): Monterey, CA 93943-5000	7b ADDRESS (City, State, and ZIP Code): Monterey, CA 93943-5000
8a NAME OF FUNDING/SPONSORING ORGANIZATION; 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO; PROJECT NO; TASK NO; WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification): NUMERICAL INVESTIGATION OF THE EFFECT OF LEADING EDGE GEOMETRY ON THE DYNAMIC STALL OF AIRFOILS

12 PERSONAL AUTHOR(S): GROHSMEYER, STEVEN P.

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM _____ TO _____	1990 September	182

16 SUPPLEMENTARY NOTATION: The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defence or the U.S. Government.

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Dynamic Stall, Oscillating Airfoil, Pitching Airfoil, Leading Edge Geometry, Pressure Gradient

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

The dynamic stall of rapidly pitching and oscillating airfoils is investigated by the numerical solution of the full compressible unsteady two-dimensional Navier-Stokes equations using an alternating-direction-implicit scheme. The flow is assumed to be fully turbulent, and the turbulent stresses are modelled by the Baldwin-Lomax eddy viscosity model. Three airfoils (NACA 0012, NACA 0012-33, and NACA 0012-63) are analyzed for the purpose of examining the influence of leading-edge geometry on unsteady flow separation. It is found that a larger leading edge radius, thicker contouring of the forward part of the airfoil, or increasing reduced frequency results in delaying flow separation and formation of the dynamic stall vortex to a higher angle of attack, yielding higher peak C_l. Within the scope of this study, the pressure gradient encountered by the flow at initial separation is found to be independent of reduced frequency and freestream speed. The critical pressure gradient is dependent on leading edge radius and increases for decreasing leading edge radius.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION	
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	Unclassified	
22a NAME OF RESPONSIBLE INDIVIDUAL: M.F. Platzer	22b TELEPHONE (Include Area Code): (408)646-2058	22c OFFICE SYMBOL: Code 31AAPL

DD Form 1473, JUN 86 — Previous editions are obsolete — SECURITY CLASSIFICATION OF THIS PAGE

S/N 0102-LF-014-6603 — UNCLASSIFIED

Approved for public release; distribution is unlimited.

Numerical Investigation of the Effect of Leading Edge Geometry on Dynamic Stall of Airfoils

by

Steven P. Grohsmeyer
Lieutenant Colonel, United States Marine Corps
B.S., Illinois Benedictine College, 1973

Submitted in partial fulfillment of the requirements for the degrees of

AERONAUTICAL ENGINEER

and

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1990

Author: Steven P. Grohsmeyer

Approved by: John. A. Ekaterinaris, Thesis Advisor

Max F. Platzer, Thesis Co-Advisor

E. Roberts Wood, Chairman,
Department of Aeronautics and Astronautics

Richard S. Elster, Dean of Instruction

ABSTRACT

The dynamic stall of rapidly pitching and oscillating airfoils is investigated by the numerical solution of the full compressible unsteady two-dimensional Navier-Stokes equations using an alternating-direction-implicit scheme. The flow is assumed to be fully turbulent, and the turbulent stresses are modelled by the Baldwin-Lomax eddy viscosity model. Three airfoils (NACA 0012, NACA 0012-33, and NACA 0012-63) are analyzed for the purpose of examining the influence of leading-edge geometry on unsteady flow separation. It is found that a larger leading edge radius, thicker contouring of the forward part of the airfoil, or increasing reduced frequency results in delaying flow separation and formation of the dynamic stall vortex to a higher angle of attack, yielding higher peak C_l. Within the scope of this study, the pressure gradient encountered by the flow at initial separation is found to be independent of reduced frequency and freestream speed. The critical pressure gradient is dependent on leading edge radius and increases for decreasing leading edge radius.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to acknowledge the instruction and support given by Dr. John Ekaterinaris in the use of advanced numerical computational facilities. In addition, his ability to link aerodynamic theory to the numerical techniques required in this investigation has been invaluable.

Likewise, over the past 18 months, Dr. Max Platzer has provided instruction and valued discussion in a number of areas of aerodynamic theory. In particular, his instruction on the full, unsteady, compressible Navier-Stokes equations and Boundary Layer theory have given me the theoretical background necessary for the completion of this study.

Finally, I would like to thank my wife Janeen for her patience and durability in typing the iterations required for this thesis.

I. INTRODUCTION

A. BACKGROUND

The aeronautical community has been aware of the phenomenon of dynamic stall for several decades. Dynamic stall is characterized by separated flow and shedding of the leading edge vortex from the upper surface of an airfoil which is rapidly pitched to angles of attack beyond the normal stalling angle of attack. The phenomenon results in significant temporary increases and decreases in lift, drag, and moment coefficients.

The scope of this work is the investigation of the phenomenon of dynamic stall for rapidly pitching and oscillating airfoils. The phenomenon of dynamic stall was observed for the first time for flows over the retreating blades of a helicopter. The dynamic stall resulting from the oscillatory motion of the rotor blade is associated with an increase in lift and the development of a severe nose down pitching moment. The effects of dynamic stall are usually undesirable for the helicopter, and where possible special care is taken to reduce its effects by special design of the rotor. On the other hand, interest has recently developed to exploit the increased lift obtained from the rapid pitch-up motion of an airfoil in order to enhance the maneuverability and extend the

flight envelope of the modern fighter aircraft to the high angle of attack regime, or to alleviate retreating blade stall for helicopters utilizing Higher Harmonic Control (HHC).

It is well known that for flow over an airfoil at fixed angle of attack the streamlined airflow is disrupted once a critical angle of attack is exceeded. At stall, the flow over the upper surface of the airfoil separates and the lift drops. It was observed, however, that rapid pitch-up motion of the airfoil delays static stall so that high lift can be maintained for angles of attack beyond the static stall angle.

As the pitch-up angle exceeds the static stalling angle of attack, a thin layer of reversed flow develops in the boundary layer. This reversed flow occurs in two types of stall: trailing edge stall and leading edge stall [Ref. 1]. In trailing edge stall, the reverse flow region begins near the trailing edge and traverses forward; in leading edge stall, the reverse flow region occurs near the suction peak just aft of the leading edge. In both cases, a vortex begins to form near the leading edge region, expands, and moves downstream. The angle of attack at which the vortex is formed depends on airfoil shape, pitch rate, mean angle and amplitude, Mach number, and Reynolds number.

The vortex formed at the leading edge is called the dynamic stall vortex, and moves with a speed of approximately 0.4 freestream speed relative to the airfoil as the pitch-up progresses. Lift, drag, and pitching moment increase signifi-

cantly until the vortex approaches the trailing edge, then drop sharply, but not simultaneously (Figure 1). The unsteady surface pressure increases, and the suction peak appears at different locations along the chord as the dynamic stall vortex moves over the airfoil. Secondary and tertiary vortices may also be present and produce additional suction peaks and fluctuations in airloads.

To date, most theoretical investigations on dynamic stall have focused on the effects of variations of reduced frequency ($k = \dot{\alpha}c/2U_\infty$), angle of attack, free stream speed, and Reynolds number on a particular airfoil shape (usually the NACA 0012 airfoil). McCroskey [Ref. 1] documented the effects of reduced frequency, amplitude and Mach number on different airfoils. These experimental studies demonstrated the large hysteresis in the

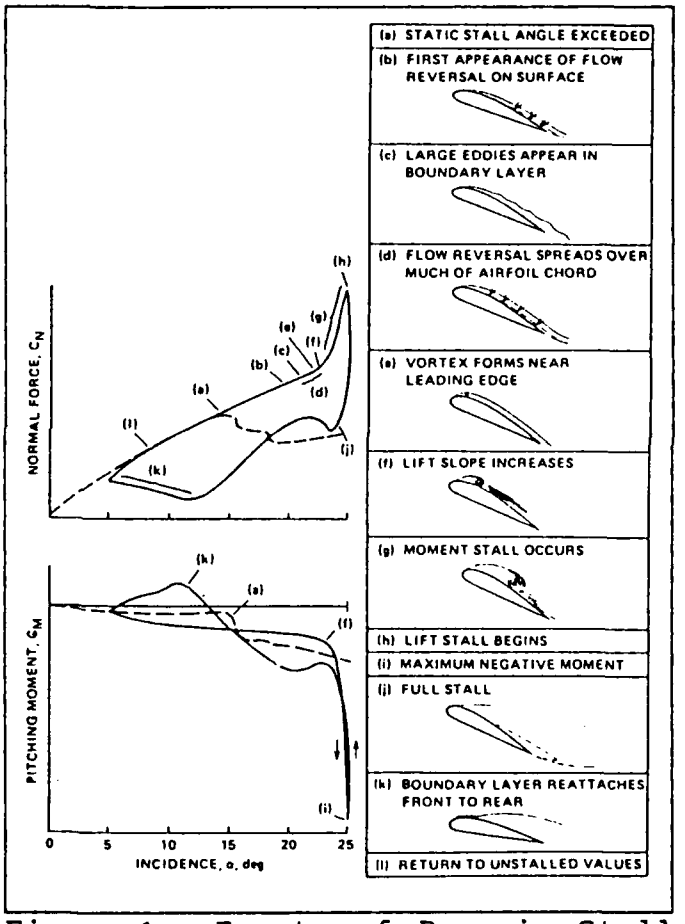

Figure 1. Events of Dynamic Stall on NACA 0012 Airfoil

occurrence of stall of an oscillating airfoil compared to a fixed angle of attack airfoil. McCroskey and Pucci [Ref. 2] identified varying regimes of viscous-inviscid interaction during varying degrees of unsteady flow separation. The conclusion of the experimental studies of Refs. 1 and 2 was that the reduced frequency has a dominant effect on the development and progression of the dynamic stall. Experimental work by Chandrasekhara and Carr [Ref. 3] using the NACA 0012 airfoil showed that a dynamic stall vortex always forms near the leading edge of an oscillating airfoil. Their study also documents the movement of the dynamic stall vortex. Chandrasekhara and Brydges [Ref. 4] documented the effects of increasing amplitude on an oscillating airfoil in both compressible and incompressible flow. They showed that larger amplitudes resulted in vortex retention at higher angles of attack for a given Mach number and reduced frequency.

Progress in computational fluid dynamics has made possible the study of dynamic stall by numerical solution of the unsteady Navier-Stokes equations. Mehta [Ref. 5] demonstrated that Navier-Stokes simulation of the unsteady incompressible flow around the airfoil in oscillatory motion can reproduce the experimentally observed results. Wu et al. [Ref. 6] presented solution procedures based on an integral formulation of the incompressible Navier-Stokes equations for the computation of unsteady flow over airfoils. Beddoes [Ref. 7] and Jang et al. [Ref. 8] presented viscous-inviscid computa-

tion methods for unsteady flows. Sankar and Tang [Ref. 9], Visbal [Ref. 10], and Ekaterinaris [Ref. 11] used ADI (Alternating Direction Implicit) numerical schemes for the solution of the compressible Navier-Stokes equations, and investigated the effects of compressibility on dynamic stall. The numerical solutions for both incompressible and compressible flows showed good agreement with experimental results, and predicted the events of dynamic stalls.

B. PURPOSE

As indicated, the aforementioned studies focused on the variation of flowfield parameters on the dynamic stall of a particular airfoil. The purpose of this study is the systematic investigation of the effect of leading edge geometry on the development of dynamic stall. It is expected that the effect of the leading edge geometry will have a significant effect on both the development of the dynamic stall vortex and its subsequent shedding. This numerical investigation is intended to provide a cost-effective means of quantifying which airfoil parameter variations should be analyzed in more expensive wind-tunnel tests. The numerical investigation of airfoil parameter variations offers the benefit of optimizing the utilization of more costly test facilities.

The investigation is conducted using a numerical solution of the full two-dimensional, Reynolds-averaged Navier-Stokes

Equations, and the Baldwin-Lomax eddy viscosity model of Ref. 16 is used to obtain the turbulent stresses.

C. AIRFOIL SELECTION

Modifications to the NACA 0012 airfoil were made forward of the point of maximum thickness (12% thick at 30%) chord by the method suggested in Ref. 12. The NACA 0012 profile was retained aft of 30% chord for the two new airfoils. In this manner, changes in the flow could be attributed directly to the changes in the forward part of the airfoil. The two airfoils consisted of the NACA 0012-63 and NACA 0012-33 airfoil section forward of 30% chord. The NACA 0012-63 has the same leading edge radius as the NACA 0012 (1.58% chord) but different curvature to the point of maximum thickness. The NACA 0012-33 has a smaller leading edge radius (0.39% chord) with curvature necessary to achieve the same 12% thickness ratio at 30% chord. Airfoil coordinates are provided in Table 1, and the resulting modified profiles are shown in Figure 2.

TABLE 1. AIRFOIL COORDINATES

X/C	N0012	N0012-63	N0012-33
.01	.01701	.01682	.01040
.02	.02360	.02320	.01550
.03	.02842	.02783	.01969
.04	.03232	.03160	.02332
.05	.03555	.03474	.02661
.06	.03840	.03750	.02962
.07	.04090	.03991	.03241
.08	.04313	.04210	.03500
.09	.04505	.04403	.03741
.10	.04681	.04580	.03966
.11	.04842	.04742	.04177
.12	.04990	.04890	.04374
.13	.05123	.05023	.04557
.14	.05242	.05147	.04734
.15	.05345	.05260	.04888
.16	.05441	.05362	.05035
.18	.05610	.05540	.05300
.20	.05742	.05686	.05515
.22	.05840	.05802	.05692
.24	.05903	.05890	.05830
.25	.05943	.05924	.05881
.26	.05964	.05952	.05925
.28	.05992	.05988	.05982
.30	.06000	.06000	.06000
.40	.05803	.05803	.05803
.50	.05294	.05294	.05294
.60	.04563	.04563	.04563
.70	.03664	.03664	.03664
.80	.02623	.02623	.02623
.90	.01448	.01448	.01448
.95	.00807	.00807	.00807
1.00	.00126	.00126	.00126

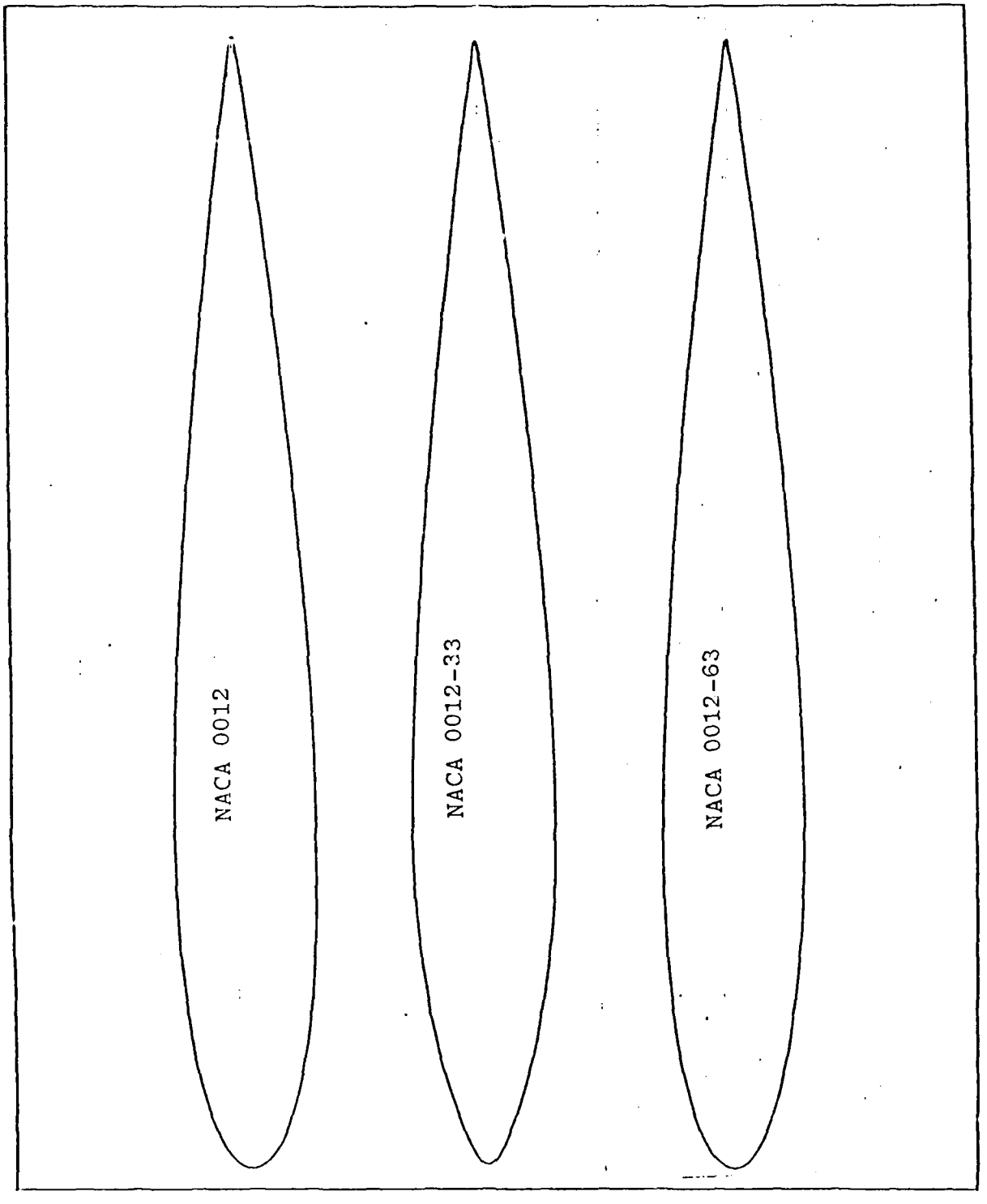

Figure 2. Airfoil Profile Comparison

II. GOVERNING EQUATIONS

The flow of a compressible, viscous fluid satisfies conservation of mass, momentum, and energy. The conservation of mass is expressed by the Continuity Equation, the conservation of momentum by the Navier-Stokes Equations, and the conservation of energy by the Energy Equation. The conservation equations are derived in the following section. The derivation is done in an Eulerian frame of reference.

A. CONTINUITY EQUATION

Consider a control volume in a flow field where flow properties vary with both time and space. Conservation of mass requires that the rate of change of mass inside the control volume equals net mass flux out of the control volume. This is expressed as:

$$\frac{\partial}{\partial t}\iiint_{vol} \rho \; d(vol) + \int_s \int \rho \vec{V} \cdot d\vec{s} = 0 \qquad (1)$$

using Gauss's theorem for a surface integral, Eq. 1 becomes

$$\iiint_{vol} \frac{\partial \rho}{\partial t} d(vol) + \iiint_{vol} \nabla \cdot (\rho \vec{V}) \, d(vol) = 0 \qquad (2)$$

or,

$$\iiint_{vol} [\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{V})] \, d(vol) = 0 \qquad (3)$$

therefore, for an arbitrary control volume,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{V}) = 0 \qquad (4)$$

for any continuous flow. For a two-dimensional flow, and Cartesian coordinates, Eq. 4 reduces to

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u)}{\partial x} + \frac{\partial (\rho v)}{\partial y} = 0 \qquad (5)$$

B. THE MOMENTUM (NAVIER-STOKES) EQUATIONS

Consider a fluid element in a rectangular Cartesian coordinate system. The stresses and pressures are shown in Figure 3, and body forces are neglected. Summation of forces in the x-direction yields:

$$\begin{aligned} F_x &= \rho \, dx\, dy\, dz \frac{Du}{Dt} \\ &= \left[p - \left(p + \frac{\partial u}{\partial x} dx\right)\right] dy\, dz + \left[\left(\tau_{xx} + \frac{\partial \tau_{xx}}{\partial x} dx\right) - \tau_{xx}\right] dy\, dz \\ &+ \left[\left(\tau_{yx} + \frac{\partial \tau_{yx}}{\partial y} dy\right) - \tau_{yx}\right] dx\, dz + \left[\tau_{zx} + \frac{\partial \tau_{zx}}{\partial z} dz\right) - \tau_{zx}\right] dx\, dy \end{aligned} \qquad (6)$$

or,

$$\rho \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} \qquad (7)$$

Similarly, summation of forces in the y- and z- directions yields:

$$\rho \frac{Dv}{Dt} = -\frac{\partial p}{\partial y} + \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{zy}}{\partial z} \qquad (8)$$

$$\rho \frac{Dw}{Dt} = -\frac{\partial p}{\partial z} + \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z} \qquad (9)$$

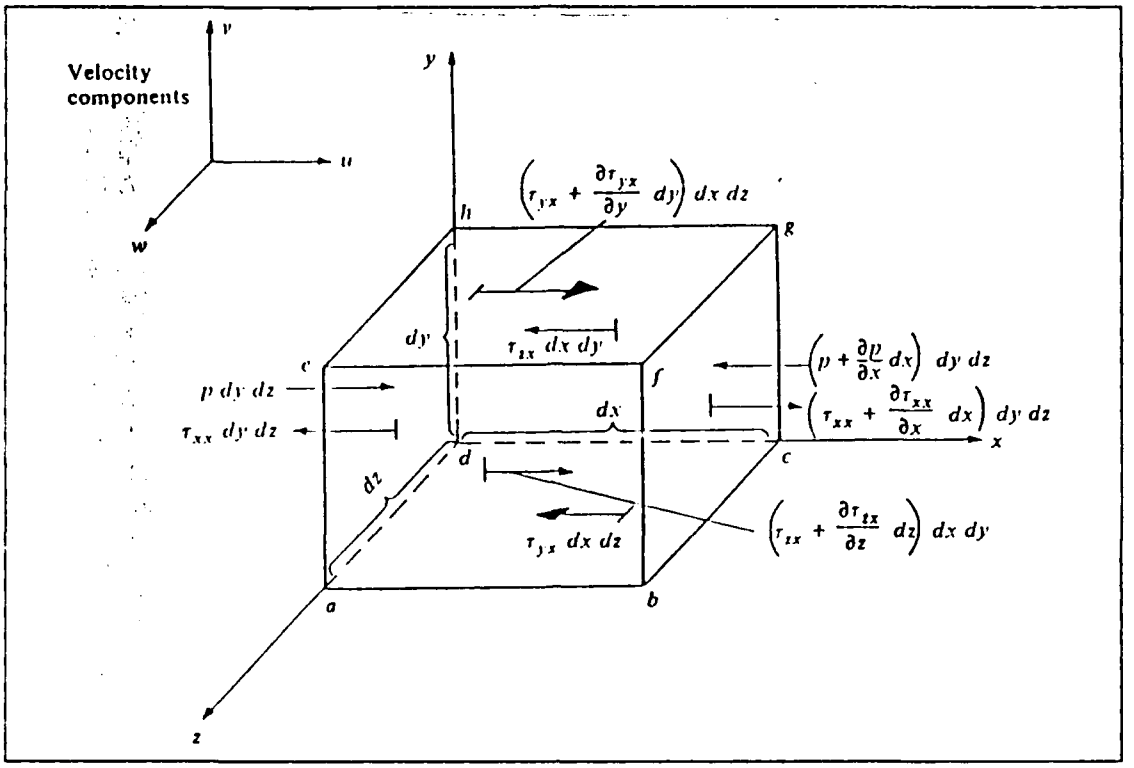

Figure 3. Forces on a Fluid Element

Using the Continuity Equation, the equations expressing the conservation of momentum for a two-dimensional flow are:

$$\frac{\partial(\rho u)}{\partial t} + \frac{\partial}{\partial x}(\rho u^2 + p) + \frac{\partial}{\partial y}(\rho uv) = \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} \qquad (10)$$

$$\frac{\partial(\rho v)}{\partial t} + \frac{\partial}{\partial x}(\rho uv) + \frac{\partial}{\partial y}(\rho v^2 + p) = \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} \qquad (11)$$

The relation of the viscous stresses τ_{xx}, τ_{yy}, and τ_{xy} to the independent variables is developed in the following discussion.

For a two-dimensional flow in Cartesian coordinates with an infinitesimal fluid element undergoing distortion due to

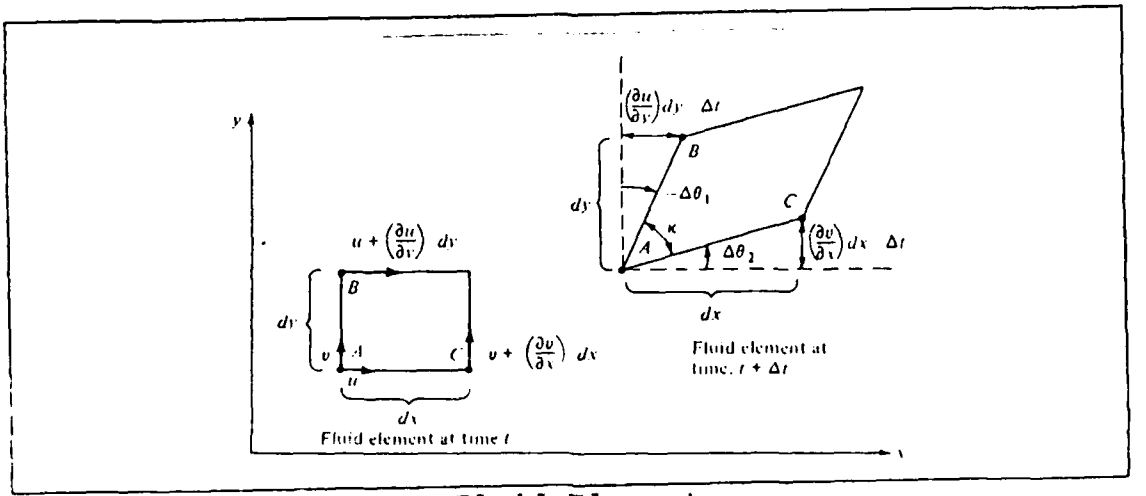

Figure 4. Strain on a Fluid Element

stresses as shown in Figure 4, the angular displacements $\Delta\theta_1$ and $\Delta\theta_2$ are:

$$\Delta\theta_2 = \frac{\partial v}{\partial x}\Delta t \qquad \Delta\theta_1 = -\frac{\partial u}{\partial y}\Delta t \tag{12}$$

The strain increment is given by:

$$\Delta K = \Delta\theta_2 - \Delta\theta_1 , \tag{13}$$

in the limit, the rate of strain is given by:

$$\varepsilon_{xy} = \frac{dk}{dt} = \frac{d\theta_2}{dt} - \frac{d\theta_1}{dt} = \frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \tag{14}$$

Using Newton's Law of Fluid Friction (definition of viscosity),

$$\tau_{ij} = \mu \frac{\partial u_i}{\partial x_j} \tag{15}$$

then the rate of strain caused by the tangential shear stress is given by:

$$\tau_{xy} = \mu \varepsilon_{xy} = \mu \left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right) \tag{16}$$

For large velocity gradients the normal stresses τ_{xx} and τ_{yy} can be significant and result in a viscous-induced normal force on the fluid element. For example, as documented by Schlichting [Ref. 13], in order for fluid isotropy to be maintained at every point, the principal axes of stress and rate-of-strain must coincide to avoid introducing a preferred rotation direction. With this concept, a normal stress must depend both on its respective component rate of strain as well as the shearing strain rates, with different weighting factors. Choosing the factor 2μ for the component direction factor, which causes Newton's Law of Friction to be satisfied for simple shear, obtain

$$\tau_{xx} = \lambda \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) + 2\mu \frac{\partial u}{\partial x} \tag{17}$$

$$\tau_{yy} = \lambda \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) + 2\mu \frac{\partial v}{\partial y} \tag{18}$$

where λ is the shearing stress proportionality factor. Using Stokes's hypothesis, $\lambda = -2/3\ \mu$, obtain

$$\tau_{xx} = \frac{2}{3}\mu\left(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}\right)$$

$$\tau_{yy} = \frac{2}{3}\mu\left(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x}\right) \qquad (19)$$

$$\tau_{xy} = \mu\left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\right)$$

C. THE ENERGY EQUATION

Conservation of energy is the manifestation of the First Law of Thermodynamics (dE = dW + dQ) to a moving fluid element in rectangular Cartesian coordinates. The First Law of Thermodynamics is applied to a control volume (Figure 5), where the energy fluxes are shown. The rate of change of energy inside the fluid element is equal to the net flux of heat into the element plus the rate of work done on the element by pressure and viscous stress.

The rate of change of energy of the fluid element having an instantaneous internal energy per unit mass e and speed V is

$$\rho\frac{\partial}{\partial t}\left(e + \frac{V^2}{2}\right)dxdydz \qquad (20)$$

where $V^2 = u^2 + v^2 + w^2$. The heat flux into the fluid element is the sum of external volumetric heating and heat transfer across the surface due to temperature gradients. Assuming no external heat addition, the volumetric heating is zero.

The net heat transferred out of the fluid element due to thermal conduction in the x-direction can be expressed as:

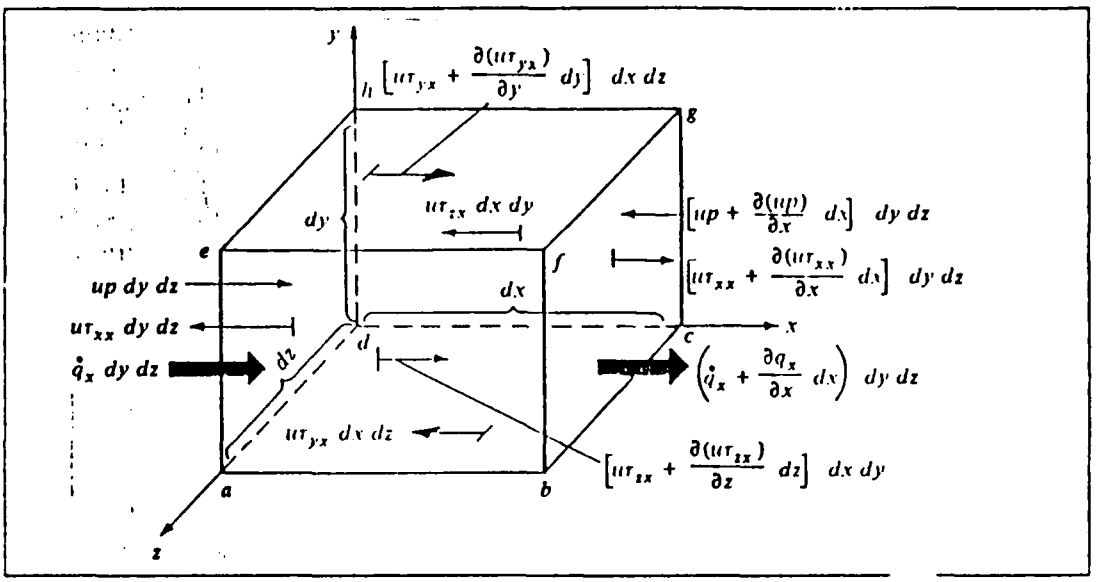

Figure 5. Energy Fluxes on Fluid Element

$$[\dot{q}_x - (\dot{q}_x + \frac{\partial \dot{q}_x}{\partial x} dx)]\, dy\, dz = -\frac{\partial \dot{q}_x}{\partial x} dx\, dy\, dz \qquad (21)$$

Accounting for the y and z directions, the total heat transferred out of the fluid element is:

$$-(\frac{\partial \dot{q}_x}{\partial x} + \frac{\partial \dot{q}_y}{\partial y} + \frac{\partial \dot{q}_z}{\partial z})\, dx\, dy\, dz = -(\nabla \cdot \vec{q})\, dx\, dy\, dz \qquad (22)$$

where the heat flux is expressed in terms of the temperature gradient according to Fourier's law of heat conduction as

$$\dot{q}_i = -k \frac{\partial T}{\partial x_i} \qquad (23)$$

and k is the thermal conductivity, considered constant and independent of temperature.

The rate of work done on the fluid element in the x-direction by pressure and shear stresses is

$$[-\frac{\partial(up)}{\partial x} + \frac{\partial(u\tau_{xx})}{\partial x} + \frac{\partial(u\tau_{yx})}{\partial y} + \frac{\partial(u\tau_{zx})}{\partial z}]\, dx\,dy\,dz \qquad (24)$$

Accounting for the y and z direction pressure and shear stresses, the net rate of work done on the fluid element can be expressed as:

$$\begin{aligned}[-\nabla\cdot p\vec{V} &+ \frac{\partial(u\tau_{xx})}{\partial x} + \frac{\partial(u\tau_{yx})}{\partial y} + \frac{\partial(u\tau_{zx})}{\partial z} \\ &+ \frac{\partial(v\tau_{xy})}{\partial x} + \frac{\partial(v\tau_{yy})}{\partial y} + \frac{\partial(v\tau_{zy})}{\partial z} \\ &+ \frac{\partial(w\tau_{xz})}{\partial x} + \frac{\partial(w\tau_{yz})}{\partial y} + \frac{\partial(w\tau_{zz})}{\partial z}]\, dx\,dy\,dz\end{aligned} \qquad (25)$$

The complete energy equation for a viscous flow with no external heating can then be expressed as:

$$\begin{aligned}\rho\frac{d}{dt}(e+\tfrac{1}{2}V^2) = [&-\nabla\cdot\vec{q} - \nabla\cdot p\vec{V} \\ &+ \frac{\partial(u\tau_{xx})}{\partial x} + \frac{\partial(u\tau_{yx})}{\partial y} + \frac{\partial(u\tau_{zx})}{\partial z} \\ &+ \frac{\partial(v\tau_{xy})}{\partial x} + \frac{\partial(v\tau_{yy})}{\partial y} + \frac{\partial(v\tau_{zy})}{\partial z} \\ &+ \frac{\partial(w\tau_{xz})}{\partial x} + \frac{\partial(w\tau_{yz})}{\partial y} + \frac{\partial(w\tau_{zz})}{\partial z}]\, dx\,dy\,dz\end{aligned} \qquad (26)$$

The two-dimensional compressible viscous flow energy equation reduces to:

$$\begin{aligned}&\rho\frac{d}{dt}(e+\frac{1}{2}V^2) + \frac{\partial}{\partial x}(pu) + \frac{\partial}{\partial y}(pv) \\ &= \frac{\partial}{\partial x}(u\tau_{xx} + v\tau_{xy} - \dot{q}_x) + \frac{\partial}{\partial y}(u\tau_{yx} + v\tau_{yy} - \dot{q}_y)\end{aligned} \qquad (27)$$

Using the continuity equation and setting

$$E = (e + \tfrac{1}{2}V^2)\rho \ , \tag{28}$$

the total energy per unit volume, the energy equation becomes

$$\begin{aligned} &\frac{\partial E}{\partial t} + \frac{\partial}{\partial x}[(E+p)u] + \frac{\partial}{\partial y}[(E+p)v] \\ &= \frac{\partial}{\partial x}(u\tau_{xx} + v\tau_{xy} - \dot{q}_x) + \frac{\partial}{\partial y}(u\tau_{xy} + v\tau_{yy} - \dot{q}_y) \end{aligned} \tag{29}$$

where τ_{xx}, τ_{yy}, and τ_{xy} are as previously derived (see Eq. 19).

D. CONSERVATION LAW FORM OF THE GOVERNING EQUATIONS

The conservation law form of the governing equations for the two-dimensional viscous compressible flows can be written as [Ref. 14]

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = \frac{\partial R}{\partial x} + \frac{\partial S}{\partial y} \tag{30}$$

where Q is the vector of dependent variables, F and G are the inviscid fluxes along the x and y directions, respectively, and R and S are the viscous fluxes along the x and y directions, respectively. These terms are given by:

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix} \qquad F = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (E+p)u \end{bmatrix} \qquad G = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (E+p)v \end{bmatrix}$$

$$R = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} - \dot{q}_x \end{bmatrix} \qquad S = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ u\tau_{xy} + v\tau_{yy} - \dot{q}_y \end{bmatrix}$$

Non-dimensionalizing the equations with

$$x_i^* = \frac{x_i}{L} \qquad u_i^* = \frac{u_i}{V_\infty} \qquad \rho^* = \frac{\rho}{\rho_\infty V_\infty^2} \qquad p^* = \frac{p}{\rho_\infty V_\infty^2}$$

$$T^* = \frac{T}{T_\infty} \qquad \mu^* = \frac{\mu}{\mu_\infty} \qquad e^* = \frac{e}{V_\infty^2} \qquad t^* = \frac{t}{L/V_\infty}$$

where L is the reference length, and setting

$$Re = \frac{\rho_\infty V_\infty L}{\mu_\infty} \tag{31}$$

the Reynolds number based on the reference length, obtain the non-dimensional form of the conservation law:

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = \frac{1}{Re}\left(\frac{\partial R}{\partial x} + \frac{\partial S}{\partial y}\right) \tag{32}$$

with non-dimensional variable Q and the inviscid flux terms (F and G) as before, and the viscous terms given by:

$$R = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} - \dfrac{\mu}{(\gamma-1)M^2 Pr}\dfrac{\partial T}{\partial x} \end{bmatrix} \tag{33}$$

$$S = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ u\tau_{xy} + v\tau_{yy} - \dfrac{\mu}{(\gamma-1)M^2 Pr}\dfrac{\partial T}{\partial y} \end{bmatrix} \qquad (34)$$

Here $Pr = C_p \mu/K$ is the Prandtl number, and $M = |\nabla|/a_\infty$. In order to enable solution of the governing equation for arbitrary geometries, the governing equations are expressed for a curvilinear generalized coordinate system. The curvilinear coordinate system (ξ,η) is linked to the Cartesian coordinate system (x,y) by

$$\xi = \xi(x,y,t) \ , \qquad \eta = \eta(x,y,t) \qquad (35)$$

The curvilinear coordinate space (ξ,η) is referred to as the computational domain and is linked to the physical domain (x,y) by the non-zero Jacobian of the coordinates transformation

$$J = \frac{\partial(\xi,\eta)}{\partial(x,y)} = \frac{\partial\xi}{\partial x}\frac{\partial\eta}{\partial y} - \frac{\partial\xi}{\partial y}\frac{\partial\eta}{\partial x} = \frac{1}{\dfrac{\partial x}{\partial\xi}\dfrac{\partial y}{\partial\eta} - \dfrac{\partial x}{\partial\eta}\dfrac{\partial y}{\partial\xi}} \qquad (36)$$

It can be shown that the governing equations [Ref. 18] retain the strong conservation law form for a generalized coordinate system. The two-dimensional strong conservation law form for a generalized coordinate system is:

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial\xi} + \frac{\partial G}{\partial\eta} = \frac{1}{Re}\left(\frac{\partial R}{\partial\xi} + \frac{\partial S}{\partial\eta}\right) \qquad (37)$$

where

$$Q = \frac{1}{J}\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ E\end{bmatrix} \tag{38}$$

$$F=\frac{1}{J}\begin{bmatrix}\rho U\\ \rho uU+p\frac{\partial \xi}{\partial x}\\ \rho vU+p\frac{\partial \xi}{\partial y}\\ (E+p)U-p\frac{\partial \xi}{\partial t}\end{bmatrix} \qquad G=\frac{1}{J}\begin{bmatrix}\rho V\\ \rho uV+p\frac{\partial \eta}{\partial x}\\ \rho vV+p\frac{\partial \eta}{\partial y}\\ (E+p)V-p\frac{\partial \eta}{\partial t}\end{bmatrix} \tag{39,40}$$

and U and V are the contravariant velocity components along the ξ and η directions, respectively given by:

$$U=\frac{\partial \xi}{\partial t}+u\frac{\partial \xi}{\partial x}+v\frac{\partial \xi}{\partial y} \qquad V=\frac{\partial \eta}{\partial t}+u\frac{\partial \eta}{\partial x}+v\frac{\partial \eta}{\partial y} \tag{41,42}$$

The viscous terms are given by:

$$R=\frac{1}{J}\begin{bmatrix}0\\ \tau_{xx}\frac{\partial \xi}{\partial x}+\tau_{xy}\frac{\partial \xi}{\partial y}\\ \tau_{xy}\frac{\partial \xi}{\partial x}+\tau_{yy}\frac{\partial \xi}{\partial y}\\ \left[\left(u\tau_{xx}+v\tau_{xy}-\frac{\mu}{(\gamma-1)M^2 Pr}\frac{\partial T}{\partial x}\right)\frac{\partial \xi}{\partial x}\right.\\ \left.+\left(u\tau_{xy}+v\tau_{yy}-\frac{\mu}{(\gamma-1)m^2 Pr}\frac{\partial T}{\partial y}\right)\frac{\partial \xi}{\partial y}\right]\end{bmatrix} \tag{43}$$

and

$$S = \frac{1}{J}\begin{bmatrix} 0 \\ \tau_{xx}\frac{\partial \eta}{\partial x} + \tau_{xy}\frac{\partial \eta}{\partial y} \\ \tau_{xy}\frac{\partial \eta}{\partial x} + \tau_{yy}\frac{\partial \eta}{\partial y} \\ \left[\left(u\tau_{xy} + v\tau_{yy} - \frac{\mu}{(\gamma-1)M^2 Pr}\frac{\partial T}{\partial y}\right)\frac{\partial \eta}{\partial x}\right. \\ \left. + \left(u\tau_{xy} + v\tau_{yy} - \frac{\mu}{(\gamma-1)m^2 Pr}\frac{\partial T}{\partial y}\right)\frac{\partial \eta}{\partial y}\right] \end{bmatrix} \qquad (44)$$

III. NUMERICAL APPROACH

A. NUMERICAL PROCEDURE

The strong conservation law form of the two-dimensional Continuity, Navier-Stokes, and Energy Equations in generalized coordinates presented in Chapter II provides a useable format for implementing a numerical solution technique. The numerical method used for the integration of the governing equations is a finite difference numerical scheme based on the Beam-Warming algorithm [Ref. 15]. The viscous terms are retained in both directions in order to enable capturing intense viscous effects encountered in massively separated flow regions at high angles of attack. The turbulent stresses were modeled using the Baldwin-Lomax eddy viscosity model [Ref. 16]. An algebraic C-type grid (157x58) was used for the computations. The boundary conditions were treated explicitly, and the unsteadiness was imposed by the motion of the grid.

B. THE BEAM-WARMING ALGORITHM

The strong conservation law form of the two-dimensional compressible Navier-Stokes equations in the vector notation is as follows:

$$\frac{\partial U}{\partial t} + \frac{\partial E(U)}{\partial x} + \frac{\partial F(U)}{\partial y} = \frac{\partial V_1(U, U_x)}{\partial x} + \frac{\partial V_2(U, U_y)}{\partial x} + \frac{\partial W_1(U, U_x)}{\partial y} + \frac{\partial W_2(U, U_y)}{\partial y} \tag{45}$$

here the vector U, the non-linear inviscid terms E and F, and the viscous terms R and S are given by:

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E_t \end{bmatrix} \quad \mathbf{E(U)} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (E_t + p)u \end{bmatrix} \quad \mathbf{F(U)} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (E_t + p)v \end{bmatrix}$$

$$\mathbf{R} = \mathbf{V_1} + \mathbf{V_2} = \begin{bmatrix} 0 \\ \frac{2}{3}\mu(2u_x - v_y) \\ \mu(u_y + v_x) \\ \mu v(u_y + v_x) + \frac{2}{3}\mu u(2u_x - v_y) + kT_x \end{bmatrix} \tag{46}$$

$$\mathbf{S} = \mathbf{W_1} + \mathbf{W_2} = \begin{bmatrix} 0 \\ \mu(u_y + v_x) \\ \frac{2}{3}\mu(2v_y - u_x) \\ \mu u(u_y + v_x) + \frac{2}{3}\mu v(2v_y - u_x) + kT_y \end{bmatrix}$$

The Beam-Warming numerical algorithm is an implicit finite difference scheme where the solution is marched in time using the difference formula

$$\Delta^n U = \frac{\theta_1 \Delta t}{1+\theta_2} \frac{\partial}{\partial t}(\Delta^n U) + \frac{\Delta t}{1+\theta_2} \frac{\partial}{\partial t}(U^n) + \frac{\theta_2}{1+\theta_2} \Delta^{n-1} U + O[(\theta_1 - \frac{1}{2} - \theta_2)(\Delta t)^2 + (\Delta t)^3] \tag{47}$$

where $\Delta^n U = U^{n+1} - U^n$. By substituting Eq. (45) into Eq. (47), obtain

$$\Delta^n U = \frac{\theta_1 \Delta t}{1+\theta_2}\left[\frac{\partial}{\partial x}(-\Delta^n E + \Delta^n V_1 + \Delta^n V_2) + \frac{\partial}{\partial y}(-\Delta^n F + \Delta^n W_1 + \Delta^n W_2)\right]$$
$$+ \frac{\Delta t}{1+\theta_2}\left[\frac{\partial}{\partial x}(-E^n + V_1^n + V_2^n) + \frac{\partial}{\partial y}(-F^n + W_1^n + W_2^n)\right] \qquad (48)$$
$$+ \frac{\theta_2}{1+\theta_2}\Delta^{n-1}U + O\left[\left(\theta_1 - \frac{1}{2} - \theta_2\right)(\Delta t)^2 + (\Delta t)^3\right]$$

The delta terms are linearized using truncated Taylor series expansions, so that:

$$E^{n+1} = E^n + \left(\frac{\partial E}{\partial U}\right)^n (U^{n+1} - U^n) + O[(\Delta t)^2] \qquad (49)$$

which can be rewritten as

$$\Delta^n E = [A]^n \Delta^n U + O[(\Delta t)^2] \qquad (50)$$

where $[A]$ is the flux Jacobian matrix $\partial E/\partial U$ given by

$$[A] = -\left[\begin{array}{c|c|c|c} 0 & -1 & 0 & 0 \\ \frac{3-\gamma}{2}u^2 + \frac{1-\gamma}{2}v^2 & (\gamma-3)u & (\gamma-1)v & (1-\gamma) \\ uv & -v & -u & 0 \\ \frac{\gamma E_t u}{\rho} + (1-\gamma)u(u^2+v^2) & -\frac{\gamma E_t}{\rho} + \frac{\gamma-1}{2}(3u^2+v^2) & (\gamma-1)uv & -\gamma u \end{array}\right] \qquad (51)$$

In a like manner, $\Delta^n F$ can be linearized as

$$\Delta^n F = [B]^n \Delta^n U + O[(\Delta t)^2] \qquad (52)$$

where $[B]$ is the Jacobian matrix $\partial F/\partial U$ given by

$$[B] = -\left[\begin{array}{c|c|c|c} 0 & 0 & -1 & 0 \\ uv & -v & -u & 0 \\ \frac{3-\gamma}{2}v^2 + \frac{1-\gamma}{2}u^2 & (\gamma-1)u & (\gamma-3)v & 1-\gamma \\ \frac{\gamma E_t v}{\rho} + (1-\gamma)v(u^2+v^2) & (\gamma-1)uv & -\frac{\gamma E_t}{\rho} + \frac{\gamma-1}{2}(3v^2+u^2) & -\gamma v \end{array}\right] \tag{53}$$

The viscous delta term $\Delta^n V_1(U, U_x)$ is linearized by writing

$$\begin{aligned} \Delta^n V_1 &= \left(\frac{\partial V_1}{\partial U}\right)^n \Delta^n U + \left(\frac{\partial V_1}{\partial U_x}\right)^n \Delta^n U_x + O[(\Delta t)^2] \\ &= [P]^n \Delta^n U + [R]^n \Delta^n U_x + O[(\Delta t)^2] \\ &= ([P] - [R_x])^n \Delta^n U + \frac{\partial}{\partial x}([R]^n \Delta^n U) + O[(\Delta t)^2] \end{aligned} \tag{54}$$

where $[P]$ is the Jacobian $\partial V_1/\partial U$, $[R]$ is the Jacobian $\partial V_1/\partial U_x$, and $[R_v] = \partial[R]/\partial x$. These matrices can be written as

$$[P] - [R_x] = -\frac{1}{\rho}\left[\begin{array}{c|c|c|c} 0 & 0 & 0 & 0 \\ -u\left(\frac{4}{3}\mu\right)_x & \left(\frac{4}{3}\mu\right)_x & 0 & 0 \\ -v\mu_x & 0 & \mu_x & 0 \\ -u^2\left(\frac{4}{3}\mu\right)_x - v^2\mu_x & u\left(\frac{4}{3}\mu\right)_x & v\mu_x & 0 \end{array}\right] \tag{55}$$

$$[R] = \frac{1}{\rho}\left[\begin{array}{c|c|c|c} 0 & 0 & 0 & 0 \\ -\frac{4}{3}\mu u & \frac{4}{3}\mu & 0 & 0 \\ -\mu v & 0 & \mu & 0 \\ -\left(\frac{4}{3}\mu - \frac{k}{c_v}\right)u^2 - \left(\mu - \frac{k}{c_v}\right)v^2 - \frac{k}{c_v}\frac{E_t}{\rho} & \left(\frac{4}{3}\mu - \frac{k}{c_v}\right)u & \left(\mu - \frac{k}{c_v}\right)v & \frac{k}{c_v} \end{array}\right] \tag{56}$$

The matrix for $[P]-[R_x]$ is obtained by assuming that μ and k are locally independent of U. In a like manner, $\Delta^n W_2(U,U_y)$ is linearized as

$$\Delta^n W_2 = ([Q] - [S_y])\,{}^n\Delta^n U + \frac{\partial}{\partial y}([S]\,{}^n\Delta^n U) + O[(\Delta t)^2] \qquad (57)$$

where

$$[Q] - [S_y] = -\frac{1}{\rho}\left[\begin{array}{c|c|c|c} 0 & 0 & 0 & 0 \\ -u\mu_y & \mu_y & 0 & 0 \\ -v\left(\frac{4}{3}\mu\right)_y & 0 & \left(\frac{4}{3}\mu\right)_y & 0 \\ -v^2\left(\frac{4}{3}\mu\right)_y - u^2\mu_y & u\mu_y & v\left(\frac{4}{3}\mu\right)_y & 0 \end{array}\right] \qquad (58)$$

and

$$[S] = \frac{1}{\rho}\left[\begin{array}{c|c|c|c} 0 & 0 & 0 & 0 \\ -\mu u & \mu & 0 & 0 \\ -\frac{4}{3}\mu v & 0 & \frac{4}{3}\mu & 0 \\ -\left(\frac{4}{3}\mu - \frac{k}{c_v}\right)v^2 - \left(\mu - \frac{k}{c_v}\right)u^2 - \frac{k}{c_v}\frac{E_t}{\rho} & \left(\mu - \frac{k}{c_v}\right)u & \left(\frac{4}{3}\mu - \frac{k}{c_v}\right)v & \frac{k}{c_v} \end{array}\right] \qquad (59)$$

The cross-derivative terms can be evaluated explicitly without loss of accuracy by noting that

$$\Delta^n V^2 = \Delta^{n-1} V_2 + O[(\Delta t)^2]$$
$$\Delta^n W^n = \Delta^{n-1} W_1 + O[(\Delta t)^2] \tag{60}$$

for a uniform time step Δt. By evaluating the cross-derivative terms in this manner, the block tridiagonal form of the final equations is maintained.

Substituting Eqs. (50), (52), (54), (57), and (59) into Eq. (48) yields

$$\begin{aligned}
&\left\{[I] + \frac{\theta_1 \Delta t}{1+\theta_2}\left[\frac{\partial}{\partial x}([A] - [P] + [R_x])^n - \frac{\partial^2}{\partial x^2}[R]^n\right.\right. \\
&\left.\left. + \frac{\partial}{\partial y}([B] - [Q] + [S_y])^n - \frac{\partial^2}{\partial y^2}[S]^n\right]\right\} \Delta^n \mathbf{U} \\
&= \frac{\Delta t}{1+\theta_2}\left[\frac{\partial}{\partial x}(-\mathbf{E} + \mathbf{V}_1 + \mathbf{V}_2)^n + \frac{\partial}{\partial y}(-\mathbf{F} + \mathbf{W}_1 + \mathbf{W}_2)^n\right] \\
&+ \frac{\theta_1 \Delta t}{1+\theta_2}\left[\frac{\partial}{\partial x}(\Delta^{n-1}\mathbf{V}_2) + \frac{\partial}{\partial y}(\Delta^{n-1}\mathbf{W}_1)\right] + \frac{\theta_2}{1+\theta_2}\Delta^{n-1}\mathbf{U} \\
&+ O\left[\left(\theta_1 - \frac{1}{2} - \theta_2\right)(\Delta t)^2, (\Delta t)^3\right]
\end{aligned} \tag{61}$$

where $[I]$ is the identity matrix. In Eq. (61), expressions such as

$$[\tfrac{\partial}{\partial x}([A] - [P] + [R_x])^n]\,\Delta^n U \tag{62}$$

are equivalent to

$$[\tfrac{\partial}{\partial x}([A] - [P] + [R_x])^n \Delta^n U] \tag{63}$$

The left-hand side of Eq. (60) is factorized in the following manner:

$$\left([I]+\frac{\theta_1\Delta t}{1+\theta_2}\left[\frac{\partial}{\partial x}\left([A]-[P]+[R_x]\right)^n-\frac{\partial^2}{\partial x^2}[R]^n\right]\right)$$

$$x\left([I]+\frac{\theta_1\Delta t}{1+\theta_2}\left[\frac{\partial}{\partial y}\left([B]-[Q]+[S_y]\right)^n-\frac{\partial^2}{\partial y^2}[S]^n\right]\right)\Delta^n U \qquad (64)$$

$$= LHS(Eq.60) + O[(\Delta t)^3]$$

and the final form of the Beam-Warming Algorithm becomes

$$LHS\,[Eq.(64)] = RHS\,[Eq.(61)]\ . \qquad (65)$$

The partial derivatives in the algorithm are evaluated using second-order accurate central differences.

The Beam-Warming algorithm is implemented in the following manner:

Step 1:

$$\left([I]+\frac{\theta_1\Delta t}{1+\theta_2}\left[\frac{\partial}{\partial x}\left([A]-[P]+[R_x]\right)^n-\frac{\partial^2}{\partial x^2}[R]^n\right]\right)\Delta^n U_1 = RHS[61] \qquad (66)$$

Step 2:

$$\left([I]+\frac{\theta_1\Delta t}{1+\theta_2}\left[\frac{\partial}{\partial y}\left([B]-[Q]+[S_y]\right)^n-\frac{\partial^2}{\partial y^2}[S]^n\right]\right)\Delta^n U = \Delta^n U_1 \qquad (67)$$

Step 3:

$$U^{n+1} = U^n + \Delta^n U \qquad (68)$$

In Step 1, $\Delta^n U_1$ represents the remaining terms on the left-hand side of Eq. (64). Equations (66) and (67) represent systems of equations which have a block tridiagonal structure.

For the two-dimensional compressible Navier-Stokes equations the block matrices have a dimension of 4x4. Central differencing is used for the second order space derivatives. Dissipation terms for numerical stability are added.

Warming and Beam [Ref. 15] have shown that the algorithm can be simplified by assuming that μ is locally constant so that $\partial\mu/\partial x=0$, $\partial\mu/\partial y=0$. Then $[P]-[R_x]=0$ and $[Q]-[S_y]=0$. Therefore, the algorithm for $\theta_1 = \frac{1}{2}$, $\theta_2 = 0$, implying second order accuracy in time, obtains the following form:

$$[I] + \frac{\Delta t}{2}\left[\frac{\partial}{\partial x}[A]^n - \frac{\partial}{\partial x}[R_x]^n\right]\Delta^n U_1 = RHS \tag{69}$$

$$[I] + \frac{\Delta t}{2}\left[\frac{\partial}{\partial x}[B]^n - \frac{\partial}{\partial y}[S_y]^n\right]\Delta^n U = \Delta^n U_1 \tag{70}$$

In the present work the viscous terms were treated explicitly in order to avoid the expensive computation of the matrices R_x and S_y. The Beam-Warming algorithm with explicit treatment of viscous terms in generalized coordinates (ξ,η) is written as

$$\begin{aligned}\left([I] + \frac{\Delta t}{2}\left[\frac{\partial}{\partial\xi}A^n + \frac{\partial}{\partial\eta}B^n\right]\right)\Delta U^{n+1} = \\ \Delta t\left(-\frac{\partial}{\partial\xi}F^n - \frac{\partial}{\partial\eta}G^n + \frac{\partial}{\partial\xi}R^n + \frac{\partial}{\partial\eta}S^n\right) = RHS^n\end{aligned} \tag{71}$$

The algorithm is further simplified by approximately factorizing the LHS(61) or LHS(71) operator in order to avoid integration of the full two-dimensional operator. The factored form of the algorithm is

$$\begin{aligned}
\left([I]+\tfrac{\Delta t}{2}\delta_\xi A^n_{i,k}\right)\Delta\bar{U}^{n+1}_{i,k} &= RHS(61)^n \\
\left([I]+\tfrac{\Delta t}{2}\delta_\eta B^n_{i,k}\right)\Delta U^{n+1}_{i,k} &= \Delta\bar{U}^{n+1}_{i,k}
\end{aligned} \tag{72}$$

In practice, implicit algorithms have stability limits due to nonlinearities. In addition, whenever discrete methods are used to compute high Reynolds number viscous behavior, small scales of motion appear which cannot be resolved by the numerics. These scales are brought about by the nonlinear interactions in the convective terms of the momentum equations. In any finite discrete mesh the small scales which cannot be resolved, result eventually in inaccuracy and contamination of the long wavelength, large scale phenomena. In order to dampen the high frequency numerical effects caused by the poor resolution of the small scales, implicit and explicit numerical dissipation is added to complete the algorithm. The numerical dissipation terms introduce an error level that does not interfere with the accuracy and resolution of any physical effects. The dissipation terms used in the present work are the ones suggested by Ref. 15. The complete factorized form of the numerical algorithms with the implicit and explicit dissipation terms is

$$\begin{aligned}
&\left[I+\left(\tfrac{\Delta t}{2}\right)\left(\partial_\xi A^n+(D_{impl})_\xi\right)\right]\times\left[I+\left(\tfrac{\Delta t}{2}\right)\left(\partial_\eta B^n+(D_{impl})_\eta\right]\Delta^n Q \\
&\quad=\Delta t\left(-\partial_\xi F^n-\partial_\eta G^n+\partial_\eta R^n+\partial_\eta S^n-\varepsilon_{expl}D_n\right)
\end{aligned} \tag{73}$$

C. THE BALDWIN-LOMAX EDDY-VISCOSITY MODEL

The Baldwin-Lomax model [Ref. 16] is an algebraic eddy viscosity model for calculating two- and three-dimensional separated flows. As opposed to the classical boundary-layer approximation which assumes zero normal pressure gradient in the boundary layer, thereby neglecting the normal momentum equation, the Baldwin-Lomax Thin Layer model retains the momentum equations and makes no assumptions about the pressure gradient. The advantage of this model arises in application to high Reynolds number, separated turbulent flows, including reverse flow regions. The Baldwin-Lomax model is a two-layer algebraic eddy-viscosity model and avoids the difficulty of finding the edge of the boundary layer. The effects of turbulence are simulated in terms of an eddy viscosity coefficient μ_t, where $\mu+\mu_t$ replaces μ (the physical viscosity coefficient) in the stress terms of the governing equations. In the model, μ_t is given by

$$\mu_t = \begin{cases} (\mu_t)_{inner} & y \le y_c \\ (\mu_t)_{outer} & y_c > y \end{cases} \tag{74}$$

where y is the normal distance from the wall and y_c is the smallest y where inner and outer values are equal. In the inner region, the Prandtl-Van Driest formula is used:

$$(\mu_t)_{inner} = \rho l^2 |\omega| \tag{75}$$

where $l = ky[1 - \exp(-y^+/A^+)]$ and $|\omega|$ is the magnitude of the vorticity. In two dimensions,

$$|\omega| = \left(\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right) \tag{76}$$

and

$$y^+ = \frac{\rho_w u_\tau y}{\mu_w} = \frac{\sqrt{\rho_w \tau_w}\, y}{\mu_w} \tag{77}$$

The subscript w indicates wall values.

For the outer region

$$(u_t)_{outer} = KC_{cp}\rho F_{WAKE} F_{KLEB}(y) \tag{78}$$

where K, C_{cp} are constants and

$F_{WAKE} = y_{MAX} F_{MAX}$ for boundary layers, or

$F_{WAKE} = C_{wk} y_{MAX} u^2_{DIF}/F_{MAX}$ for wakes and separated boundary layers.

The quantities y_{MAX} and F_{MAX} are determined from

$$F(y) = y|w|[1-\exp(y^+/A^+)] \tag{79}$$

The exponential term in this equation is set equal to zero for wakes.

F_{MAX} is the maximum values of F(y), which occurs at a value of $y = y_{MAX}$. The function $F_{KLEB}(y)$ is the Klebanoff intermittency factor given by

$$F_{KLEB}(y) = [1+5.5(\frac{C_{KLEB}y}{y_{MAX}})^6]^{-1} \tag{80}$$

The quantity U_{DIF} is the difference between $U = U|_{y=yMAX}$ and minimum total velocity at a fixed flow-wise station:

$$U_{DIF} = (\sqrt{u^2+v^2})_{y=y_{max}} - (\sqrt{u^2+v^2})_{min} \qquad (81)$$

where the second term is zero except in wakes.

In order to achieve agreement with Cebeci [Ref. 17], the values determined for the constants are:

A^+	= 26	k	= 0.4
C_{cp}	= 1.6	K	= 0.0168
C_{KLEB}	= 0.3	Pr	= 0.72
C_{WK}	= 1.0	Pr_t	= 0.9

D BOUNDARY CONDITIONS

The following boundary conditions were used in the numerical implementation. A non-slip non-penetration boundary condition in terms of the contravariant velocity components was used on the airfoil solid boundary. The unsteady pitching motion was accomplished by rotating the grid about the quarter chord point at pitch rates determined by the desired reduced frequency. The inflow boundary was placed approximately eight chord lengths away from the body surface. Therefore, freestream conditions were specified at the grid inflow boundary. Simple first order extrapolation was used for the flow variables at the outflow boundary. For the wake, averaging of the flow variables on the upper and lower surfaces of the C-grid was used. The velocity on the surface

is given by $u_s=\dot{x}=\omega z$ and $v_s=\dot{z}=-\omega x$. The contravariant velocity components for viscous flow solutions are set equal to 0. Therefore, the non-slip condition in terms of the physical velocity components for moving grid is expressed by

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{J} \begin{bmatrix} \eta_y & -\xi_y \\ -\eta_x & \xi_x \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{z} \end{bmatrix} \quad . \quad (82)$$

E. GRID GENERATION

1. Grid Generation Methods

In order to effectively compute complex flowfields, the physical domain of interest must be discretized with a finite mesh. The requirements of an efficient numerical grid are [Ref. 18]:

1. smooth grid lines so that the transformation derivatives (metrics) are continuous.
2. grid point spacing which varies inversely with expectation of large numerical errors.
3. minimizing grid skewness to avoid large truncation errors.

Several general grid generation techniques exist; among these the most common methods are:

1. Complex Variable methods, where the transformations are at least partly analytic; this method is restricted to two-dimensions.
2. Algebraic methods, usable in two- or three-dimensions.
3. Differential Equation methods, usable in two- or three-dimensions.

The grid generation method used in this study utilized the algebraic grid generation technique because of the computational efficiency and speed it provides.

2. Algebraic Grid Generation Method

The algebraic method used employs known, easily invertible, functions to map arbitrarily shaped regions (in this case, the airfoil contour) into a simpler computational domain. The airfoil surface is unwrapped to form a simple curve in the computational plane as in Figure 6. In the computational domain lines are first drawn normal to this curve, and the grid points in the normal direction are subsequently generated. In the case of airfoil grid generation, consideration must be given for the clustering of grid points near the airfoil in order to adequately resolve the near surface viscous layers. An algebraic function is used to provide a uniform stretching normal to the airfoil in the computational plane. The resulting computational domain grid is wrapped back to the initial physical grid using inverse transformations.

The grid generation program used in this study yielded a 157x58 C-type grid. The program is listed in Appendix C. Grids for the airfoils under consideration are displayed in Figures 7-9. Figure 7 displays the global grid including wake for the NACA 0012 airfoil. Figures 8 and 9 show the body-fitted grid in more detail for the two modified airfoils. Grid clustering near the airfoils is shown for resolving the

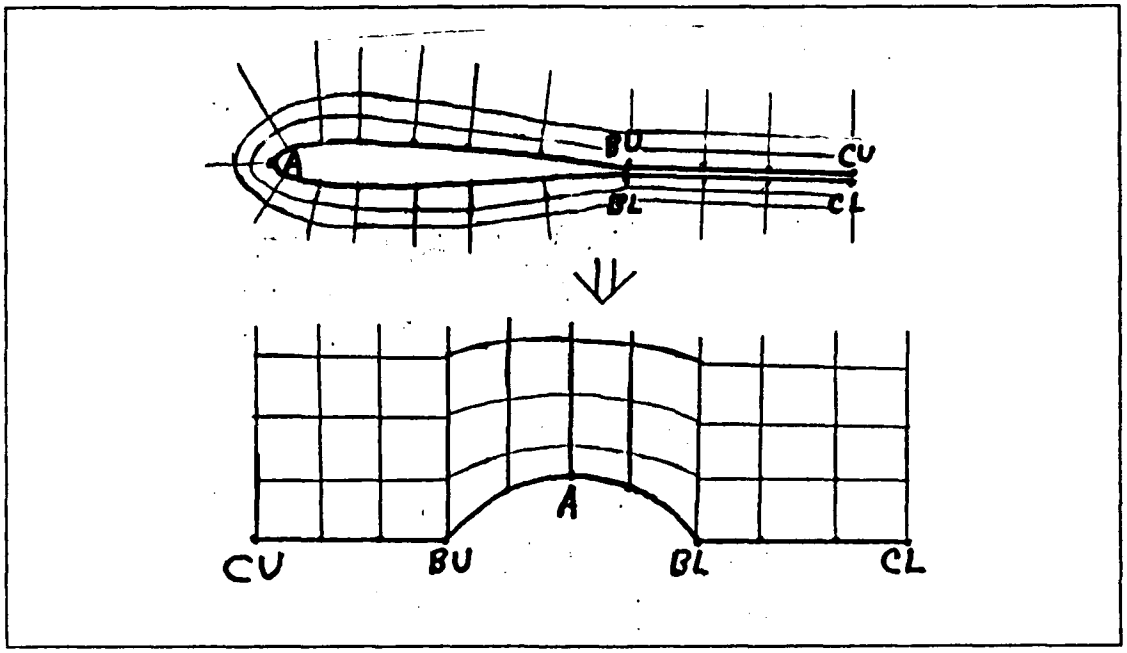

Figure 6. Airfoil Grid Unwrapping

boundary layer flow in turbulent and separated flow conditions.

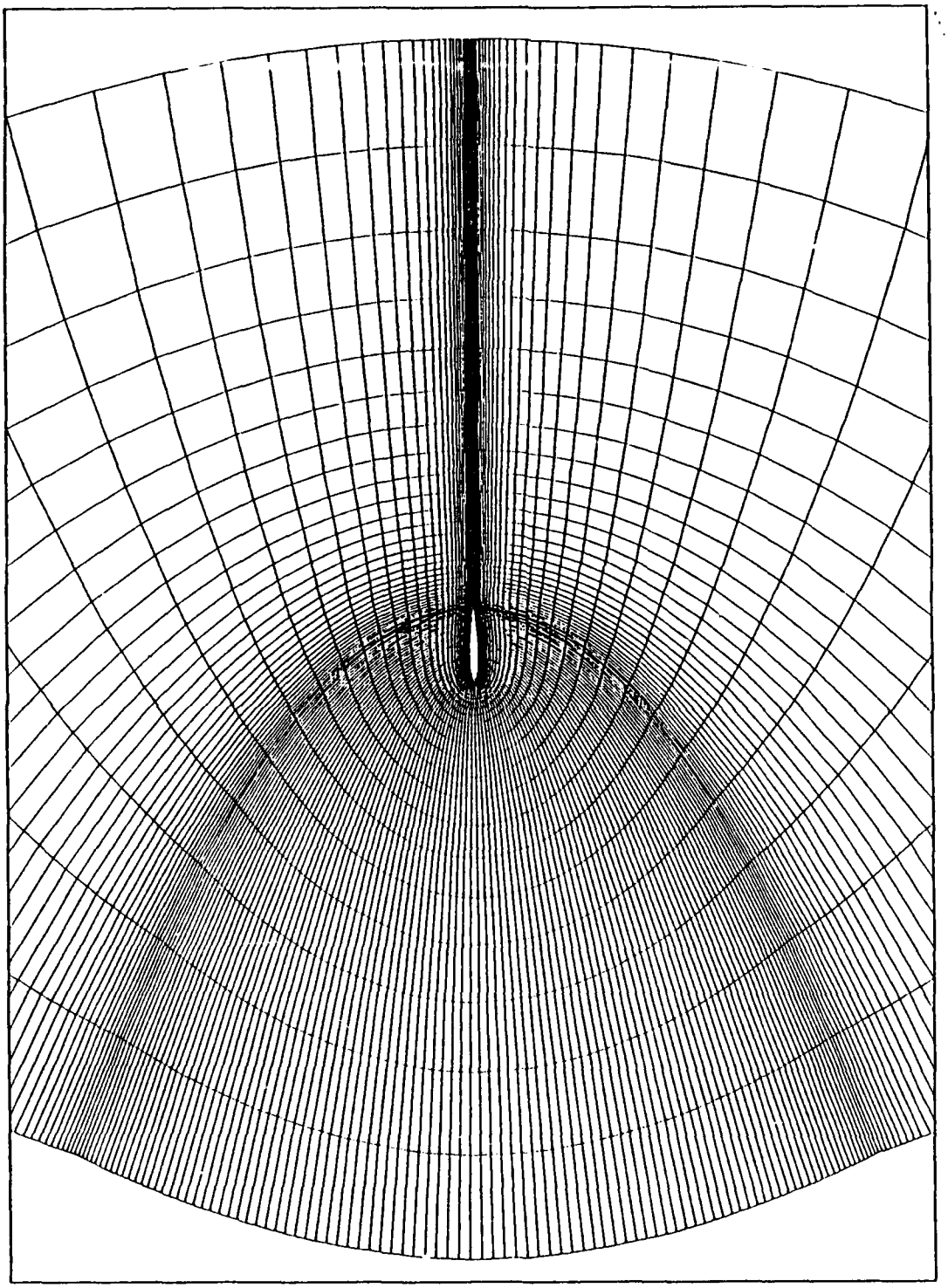

Figure 7. N0012 Global Grid

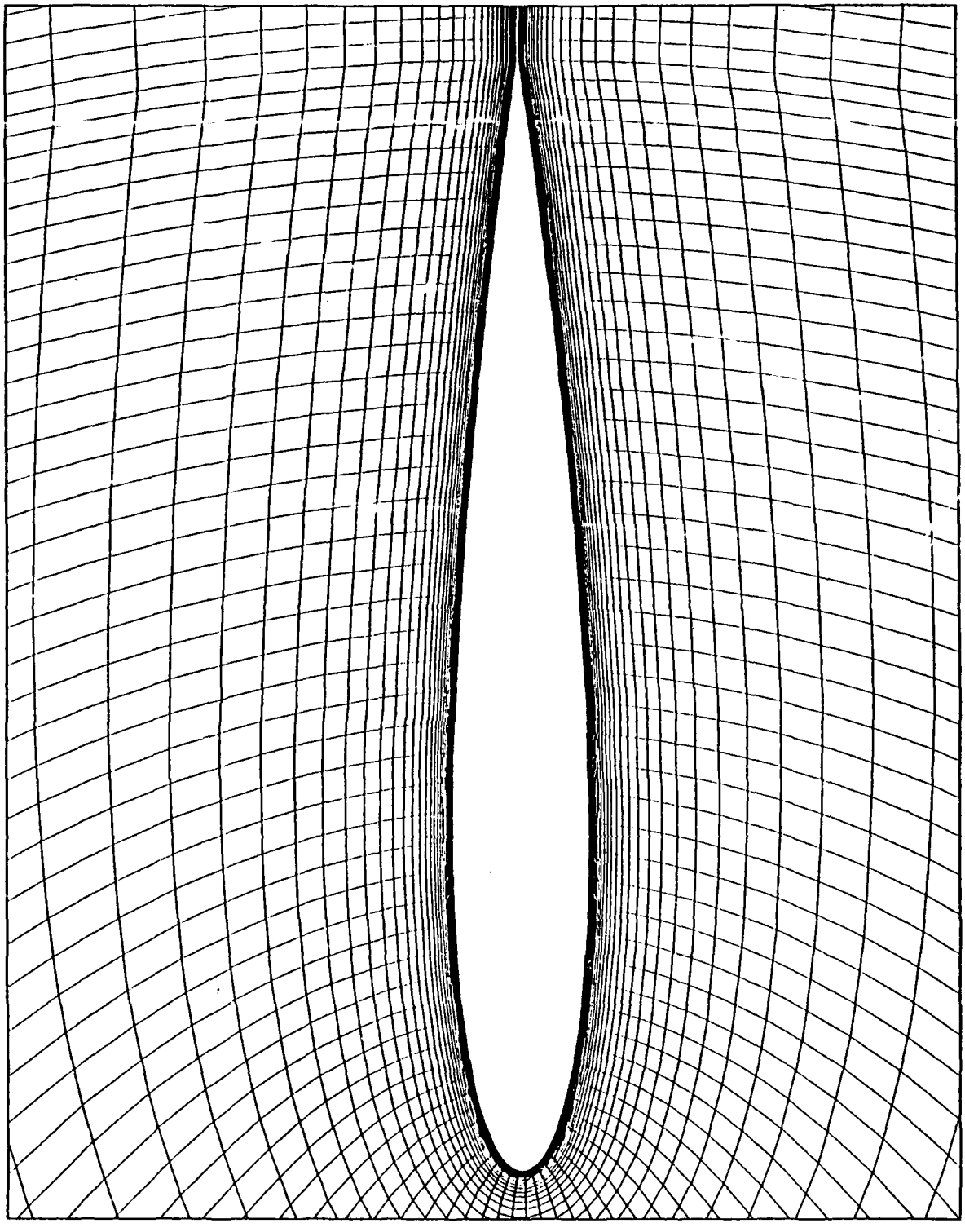

Figure 8. N0012-63 Local Grid

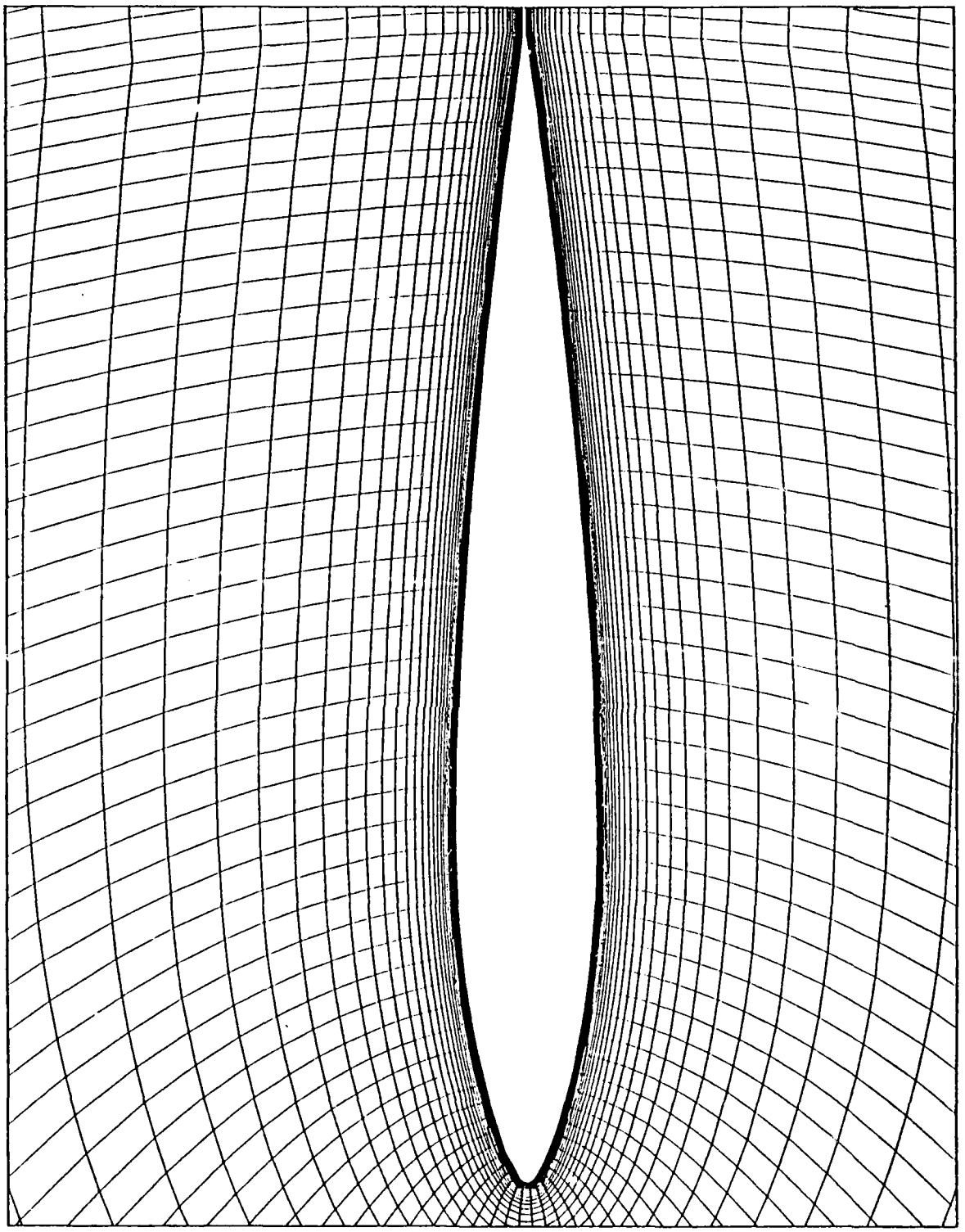

Figure 9. N0012-33 Local Grid

IV. RESULTS AND DISCUSSION

A. INVESTIGATION METHOD

Two-dimensional unsteady flows involving both harmonically oscillating and rapidly pitching (ramp pitch-up) airfoils were investigated. Studies for the oscillatory cases were conducted using the NACA 0012 airfoil in order to enable comparisons with previously conducted experimental measurements. Flows over all three airfoils rapidly pitched from 0° to 30° angle of attack with the reduced frequencies and Mach numbers (shown in Table 2) were subsequently investigated.

TABLE 2. RAPIDLY PITCHING AIRFOIL CASES, Re = 4 x 10^6

AIRFOIL	0.3 Mach		0.4 Mach	
	k=.01	k=.02	k=.01	k=.02
NACA 0012	X	X	X	X
NACA 0012-63	X	X	X	X
NACA 0012-33	X	X	X	X

Solutions for the harmonically oscillating NACA 0012 airfoil were obtained for flows at a freestream Mach number of 0.3, Reynolds number based on the root chord $Re_c=4x10^6$ and for two reduced frequencies of k=0.1 and k=0.2. The reduced frequency is defined as $k=\omega c/2U_\infty$ for the oscillatory case where $\alpha(t)=a_0 + a_1 \sin(\omega t)$. In terms of nondimensional quantities k is given by $k=\omega/2M_\infty$, and $\dot{\alpha}(t)=a_1 \omega\cos(\omega t)$ is the

instantaneous pitch-up rate which varies during the cycle with $\omega=2kM_\infty$ for a unit chord length. The present computations were conducted for a variation of the angle of attack as $\alpha(t)=10°+6\sin(t)$. Experimental test conditions for the same freestream Mach and Reynolds numbers were for $\alpha(t)=10°+5\sin(t)$. Because the airfoil stalled just at 15° for the experiment, the computed flow for the same conditions did not yield a hysteresis loop. Therefore, as McCroskey suggested, the oscillation was increased by one degree and a hysteresis loop was obtained. The computed lift behavior shown in Figures 10 and 11 exhibits the well-known hysteresis loop of a harmonically oscillating airfoil experiencing dynamic stall. The computation was initiated from a steady-state solution at $\alpha=5°$ and was carried out for two cycles. Figures 10 and 11 display the results of the second cycle from the two-cycle computation.

Flow solutions for a rapidly pitching airfoil were obtained by pitching the airfoil at a constant rate from a zero angle of attack and steady-state flow conditions to an angle of attack of 30° at the desired reduced frequency and freestream Mach number. For the case where ramp motion was imposed, the reduced frequency k is given by $k=\dot{\alpha}c/2U_\infty$ where $\dot{\alpha}$ is the constant pitch-up rate. In terms of nondimensional quantities $k=\omega/2M_\infty$, or $\omega=2kM_\infty$=constant, and $\alpha(t)=\alpha_0+(\alpha_1-\alpha_0)\omega t$, where in this study $\alpha_0=0°$ and $\alpha_1=30°$ and $\omega=\dot{\alpha}(t)$. A summary of

the computed results for the rapidly pitching airfoils is provided in Table 3.

TABLE 3. PEAK LIFT COEFFICIENTS

AIRFOIL	Mach No.	Reduced Frequency	Peak C_l	Angle of Attack
	0.3	.01	1.90	19.40°
N0012		.02	2.10	23.40°
	0.4	.01	1.97	21.08°
		.02	2.24	25.21°
	0.3	.01	1.84	18.50°
N0012-63		.02	2.09	23.00°
	0.4	.01	1.93	20.60°
		.02	2.22	25.00°
	0.3	.01	1.65	15.80°
N0012-33		.02	1.94	19.60°
	0.4	.01	1.74	17.10°
		.02	2.09	23.15°

A general observation on the computed solution is that the modified NACA 0012-63 airfoil exhibited comparable lift behavior to the baseline NACA 0012 airfoil. A slight angle of attack versus lift curve shift was observed between the two airfoils at all reduced frequency/Mach number combinations. However, for the same flow parameters the NACA 0012-33 airfoil consistently underperformed the larger leading edge radius airfoils. The difference in the unsteady lift behavior with increasing angle of attack resulted from the different flow character at the leading edge region as will be shown later in detail. In the following sections, detailed comparison of the lift behavior of the three airfoils is presented at the various flow conditions. The flow characteristics and the

development and progression of the dynamic stall process are examined. The effects of reduced frequency and freestream Mach number are also discussed.

B. LIFT BEHAVIOR

1. Harmonically Oscillating Airfoil

Although hysteresis loops characteristic of an oscillating airfoil undergoing dynamic stall were observed for reduced frequencies of both 0.1 and 0.2, agreement with experimental results varied. At a reduced frequency of 0.1, the response agreed well with McCroskey's experimental results (Figure 10) during the upstroke. Maximum lift coefficient was 1.52 at an angle of attack of 15.8°. The lift behavior continues to agree well with the experimental data during the upstroke of the hysteresis loop and during the initial part of the downstroke. As the downstroke continues and the flow reattaches, however, the numerical solution displays significantly greater oscillations in lift coefficient compared to experimental data, which were obtained as an average over several cycles of oscillation.

At a reduced frequency of 0.2 (Figure 11), the numerical solution exhibited a much smaller hysteresis loop than the respective experimental data. In this case, maximum lift coefficient in the numerical sclution occurred just prior to the downstroke, so that flow reattachment occurred almost

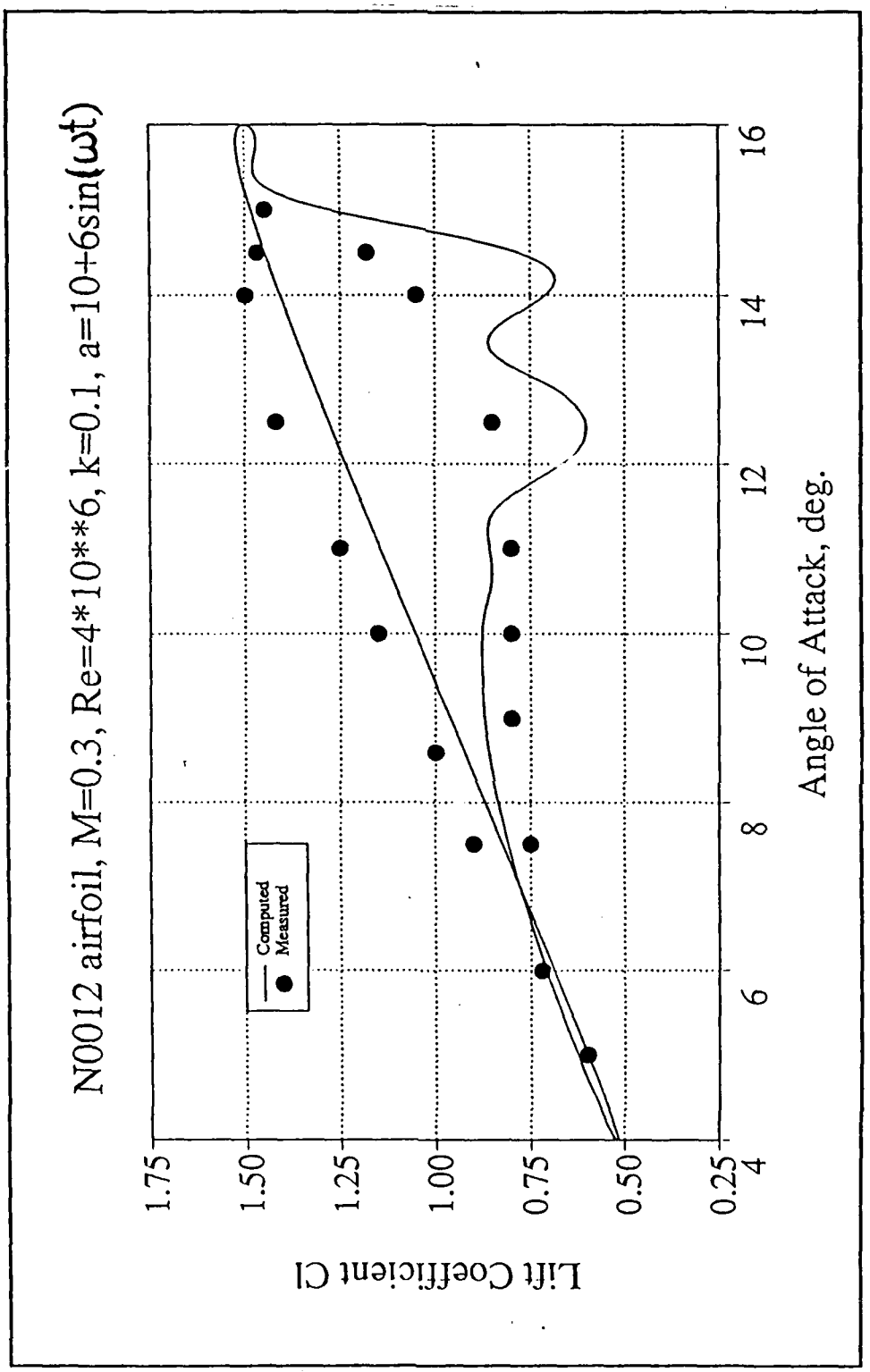

Figure 10. Lift Behavior, NACA 0012, M=0.3, k=0.1

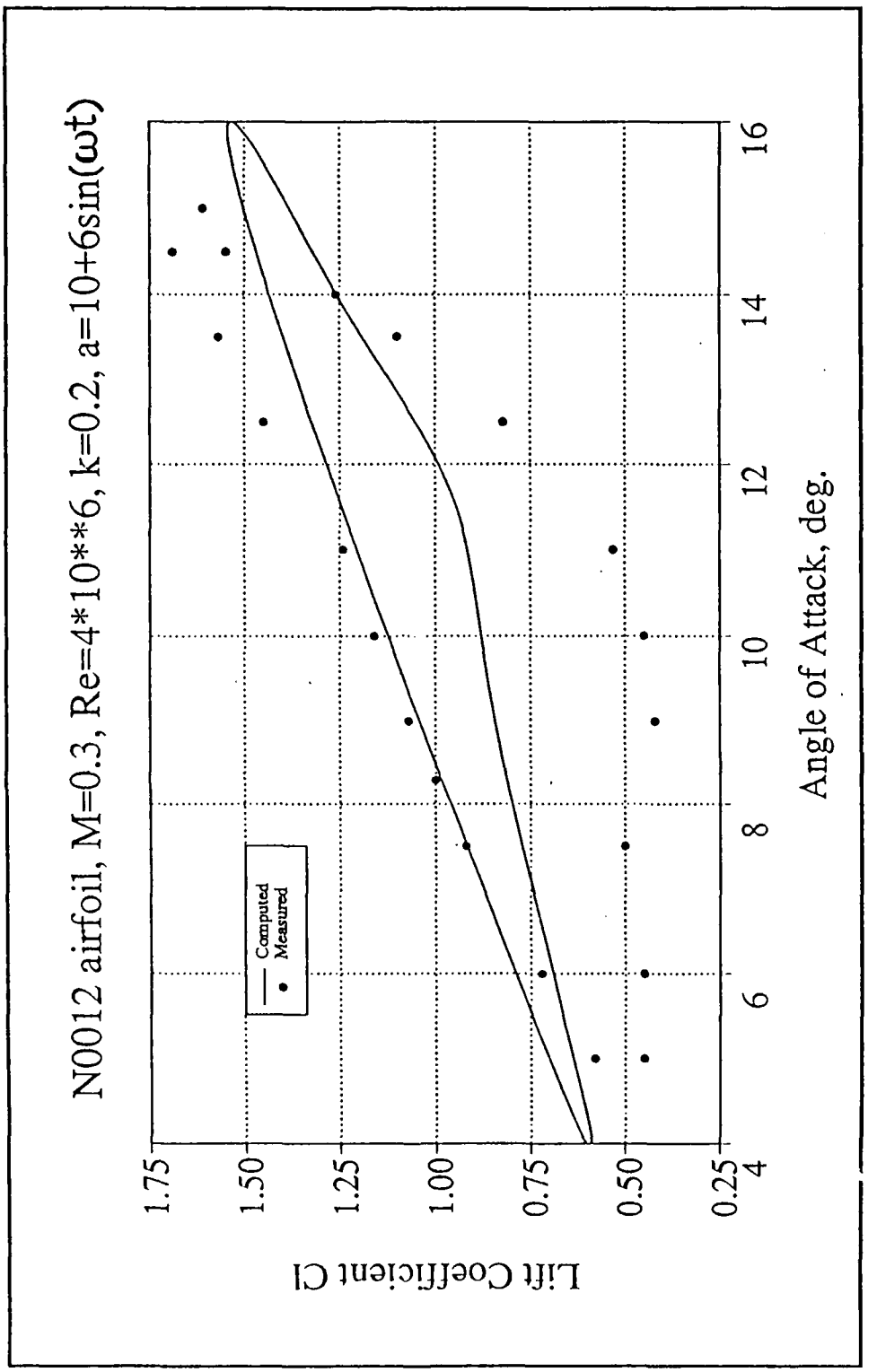

Figure 11. Lift Behavior, NACA 0012, M=0.3, k=0.2

immediately. Maximum lift coefficient at k=0.2 was only slightly higher than at the k=0.1 case.

The lack of agreement with experimental data at a reduced frequency of 0.2 and during the flow reattachment process at k=0.1 may be indicative of the poor behavior of the eddy-viscosity model (which is suitable for steady flows). Higher values of the reduced frequency resulted in larger discrepancies from the measured lift values.

2. Rapidly Pitching Airfoil

Lift coefficient vs. angle of attack comparison of the three airfoils is presented in Figures 12-15 for the Mach number/reduced frequency combinations listed in Table 2. For the same flow parameters, all three airfoils have nearly the same lift curve slope until the onset of dynamic stall. Along with the aforementioned angle of attack shift of the lift curves between the NACA 0012 and -63 airfoils after stall occurs, the peak lift coefficients for the NACA 0012 airfoil are slightly higher than for the -63 airfoil. The peak lift coefficient for the NACA 0012 occurred 0.2 to 0.9 degrees angle of attack higher than the peak C_l for the -63 airfoil for the same flow conditions. With both airfoils having the same leading edge radius, the small difference in performance may be attributed to the slightly thicker contouring of the forward part of the baseline NACA 0012 airfoil (Table 1).

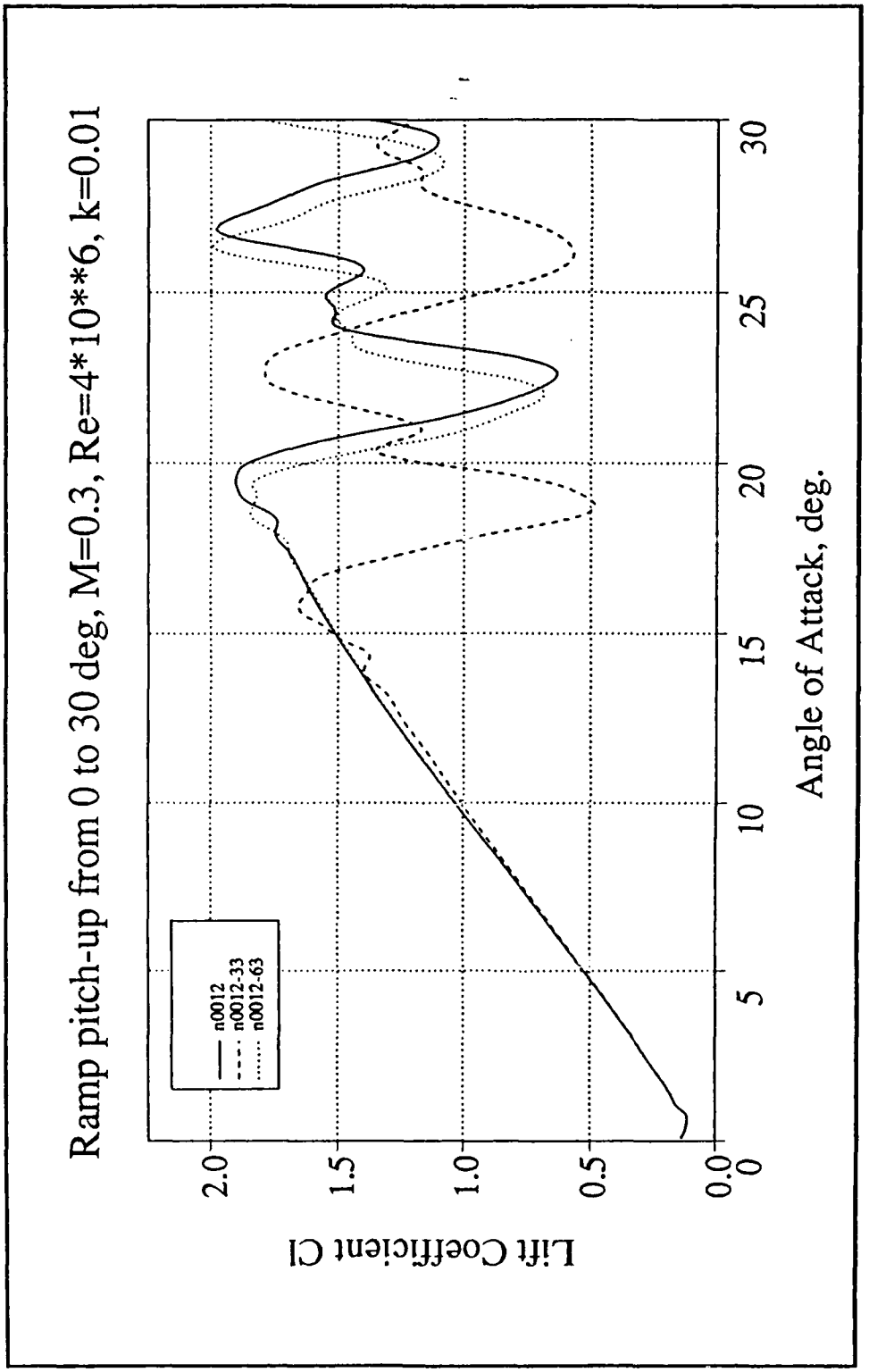

Figure 12. Lift Comparison, M=0.3, k=0.01

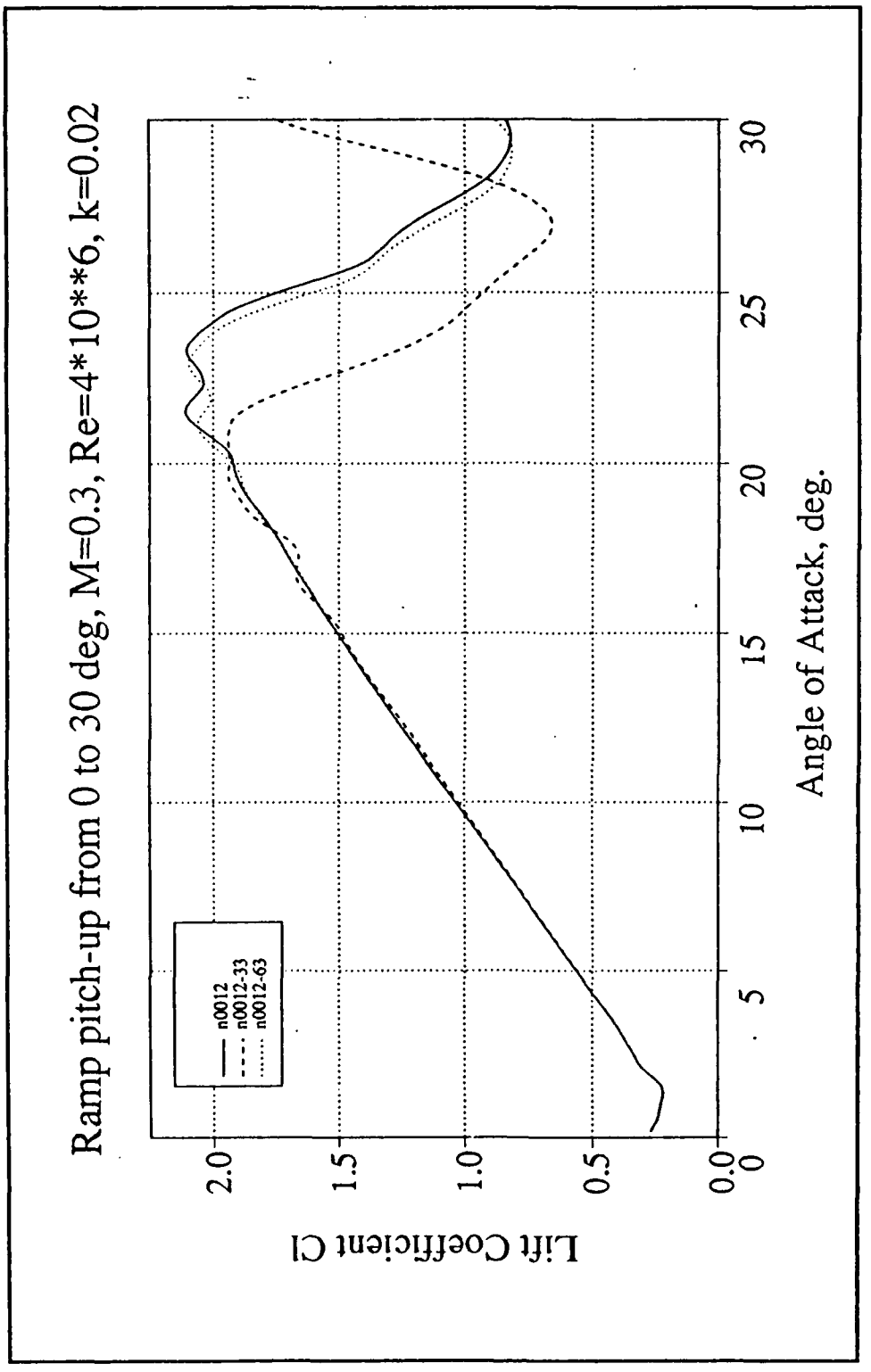

Figure 13. Lift Comparison, M=0.3, k=0.02

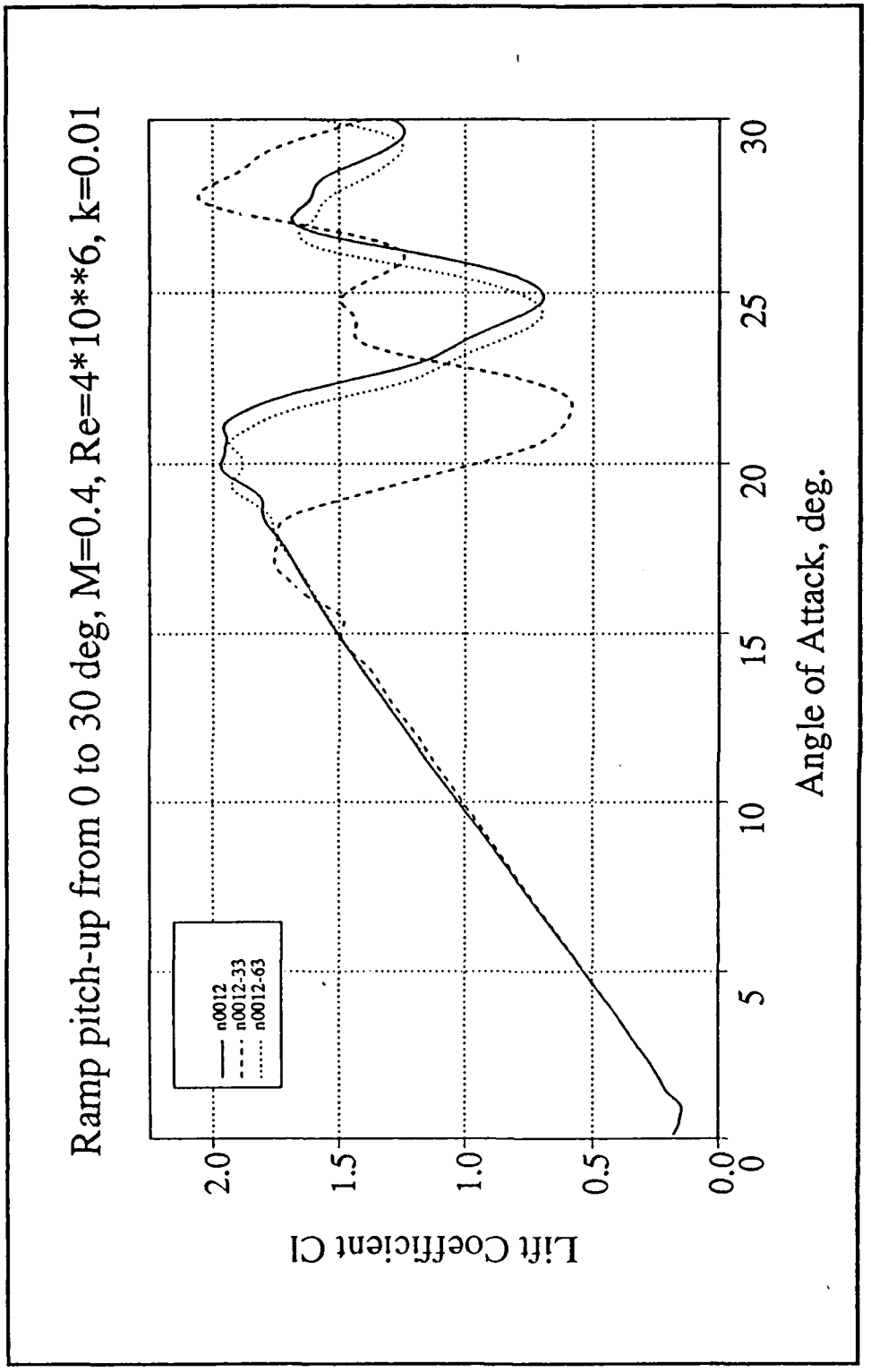

Figure 14. Lift Comparison, M=0.4, k=0.01

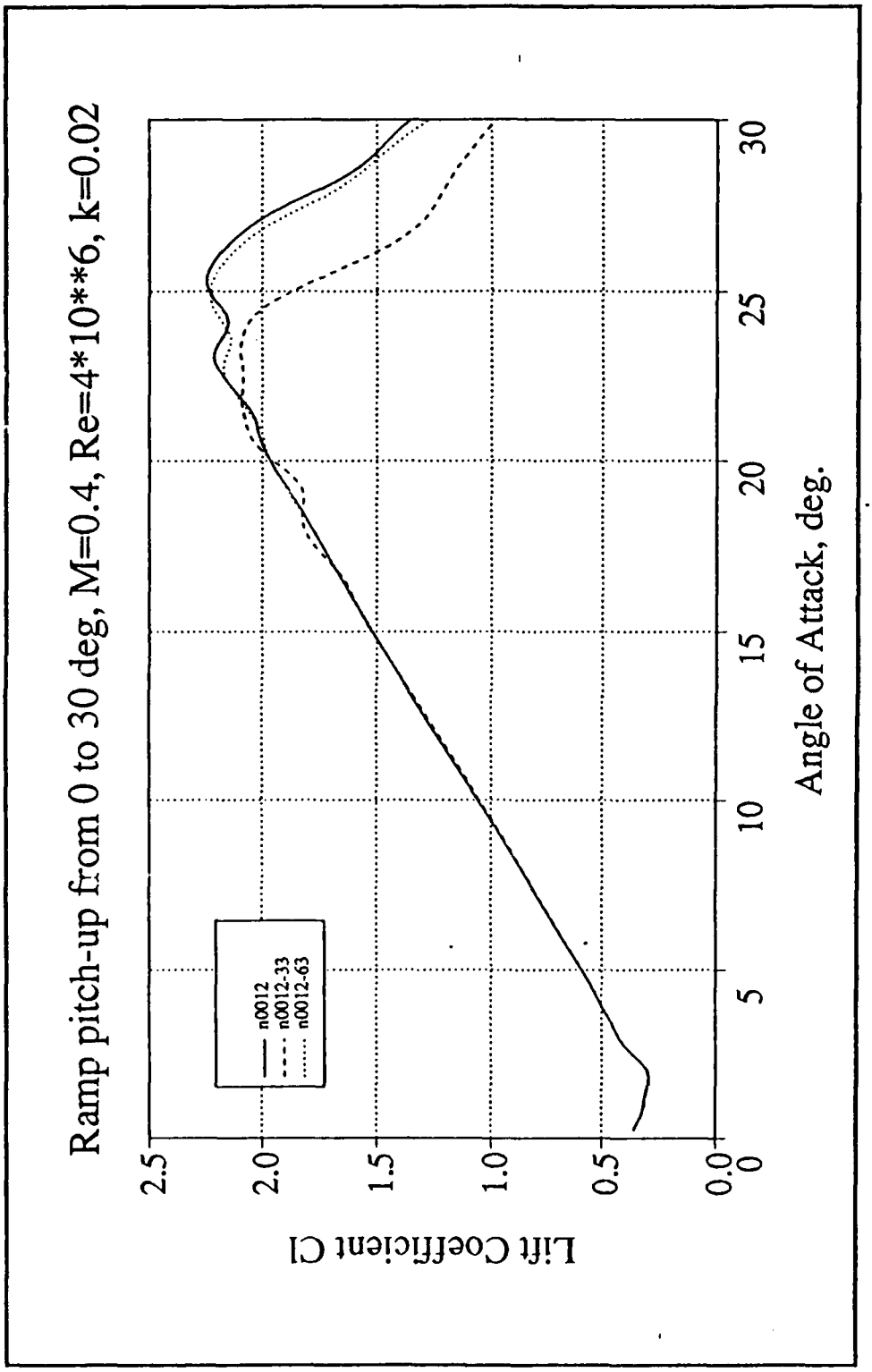

Figure 15. Lift Comparison, M=0.4, k=0.02

The maximum lift coefficient obtained with the NACA 0012-33 airfoil was lower than that obtained with the other two airfoils and occurred at 1.8 to 4.0 degrees lower angle of attack. At a reduced frequency of k=0.01 and freestream Mach number of 0.3, the initial leading edge vortex originated at an angle of attack of 12° for the -33 airfoil. Under the same flow conditions, formation of the vortex occurred at 17° and 16.5° for the baseline 0012 and -63 airfoils, respectively. Similar differences occurred at all other flow conditions investigated. The smaller leading edge radius of the NACA 0012-33 airfoil promoted earlier development of the dynamic stall vortex from the leading edge upper surface.

As will be shown, the dynamic stall vortex originates as a result of the combination of the accelerated flow over the leading edge with the boundary layer reverse flow behind the leading edge. An adverse pressure gradient is encountered by the flow as it passes the suction peak just downstream of the airfoil leading edge. This adverse pressure gradient causes flow deceleration just aft of the suction peak, leading eventually to the boundary layer separating at a critical value of the adverse pressure gradient. The momentum of the flow, however, is increased as the flow accelerates around the leading edge and opposes the tendency of the flow to separate. Eventually, depending on the airfoil shape characteristics, the boundary layer separates, reversed flow occurs, and combines with the freestream to form the leading edge vortex.

As observed in this investigation, the size of the leading edge radius is of primary importance in determining the angle of attack at which the boundary layer separates and rolls to form the dynamic stall vortex. For the same flow parameters and angle of attack during pitch-up, the critical pressure gradient for flow separation occurred earlier on the airfoil with smaller leading edge radius (NACA 0012-33). Stated otherwise, at a given angle of attack during pitch-up, the adverse pressure gradient aft of the suction peak is greater for the smaller leading edge airfoil.

The contouring of the forward part of the airfoil is of secondary importance in alleviating development of the adverse pressure gradient. The effect of contouring is indicated by the slightly earlier development of leading edge reverse flow on the NACA 0012-63 airfoil compared to the baseline 0012 airfoil.

In summary, enlarging the leading edge radius and thickening the contouring of the forward part of the airfoil results in decreasing the adverse pressure gradient encountered by the flow, with resulting delay in separation. This is shown graphically in Figures 16-20 where instantaneous streamlines are shown for the three airfoils at the same 17° angle of attack, 0.4 Mach, and k=0.01. Reversed boundary layer flow has just begun on the NACA 0012 airfoil aft of the suction peak, whereas initiation of the dynamic stall vortex has already occurred on the NACA 0012-63 airfoil which has

thinner contouring. The NACA 0012-33 airfoil with the small leading edge radius has a fully developed dynamic stall vortex and is 0.6° angle of attack prior to dynamic stall. A detailed discussion of the vortical flow development follows.

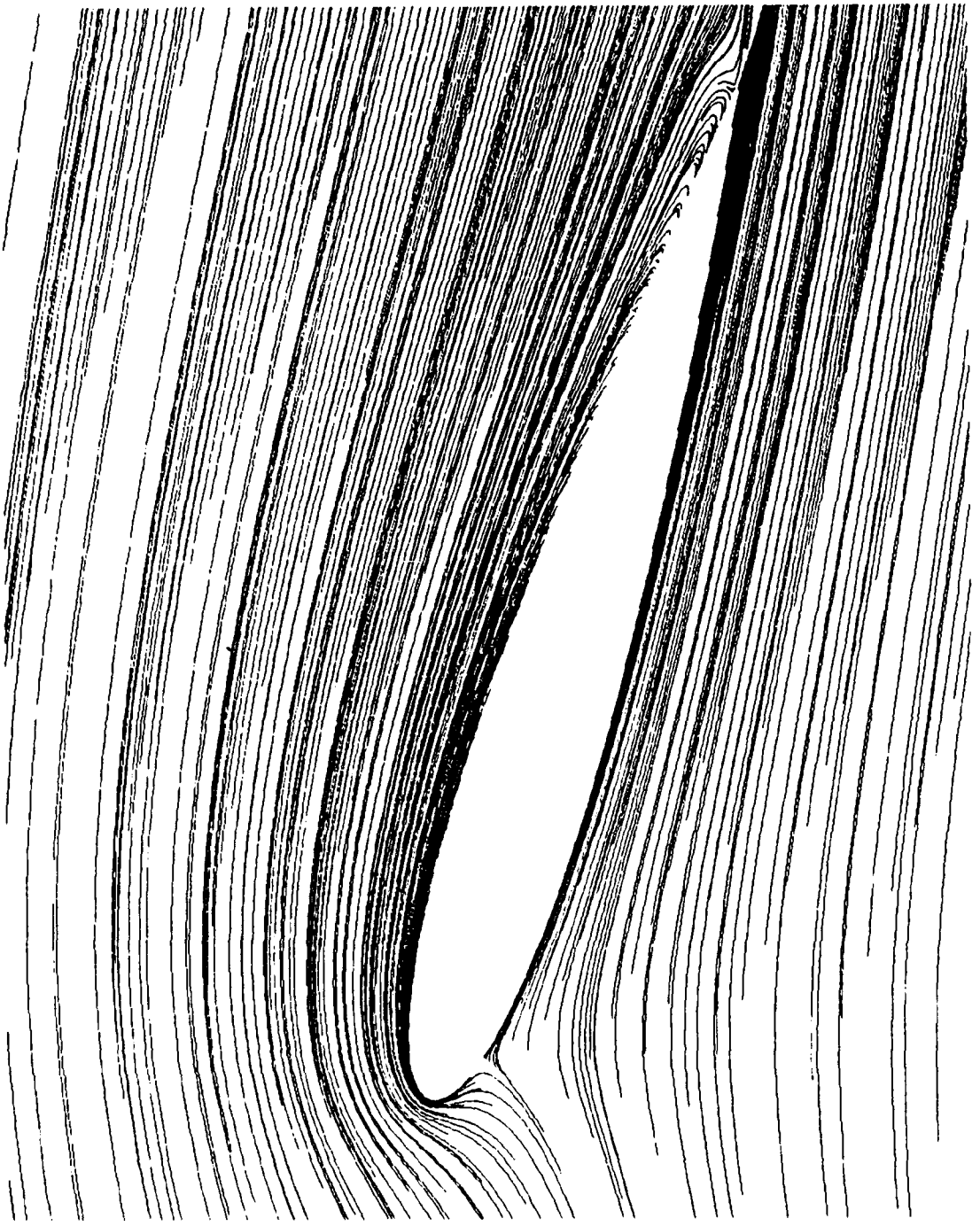

Figure 16. Instantaneous Streamlines, NACA 0012, M=0.4, k=0.01, α=17°

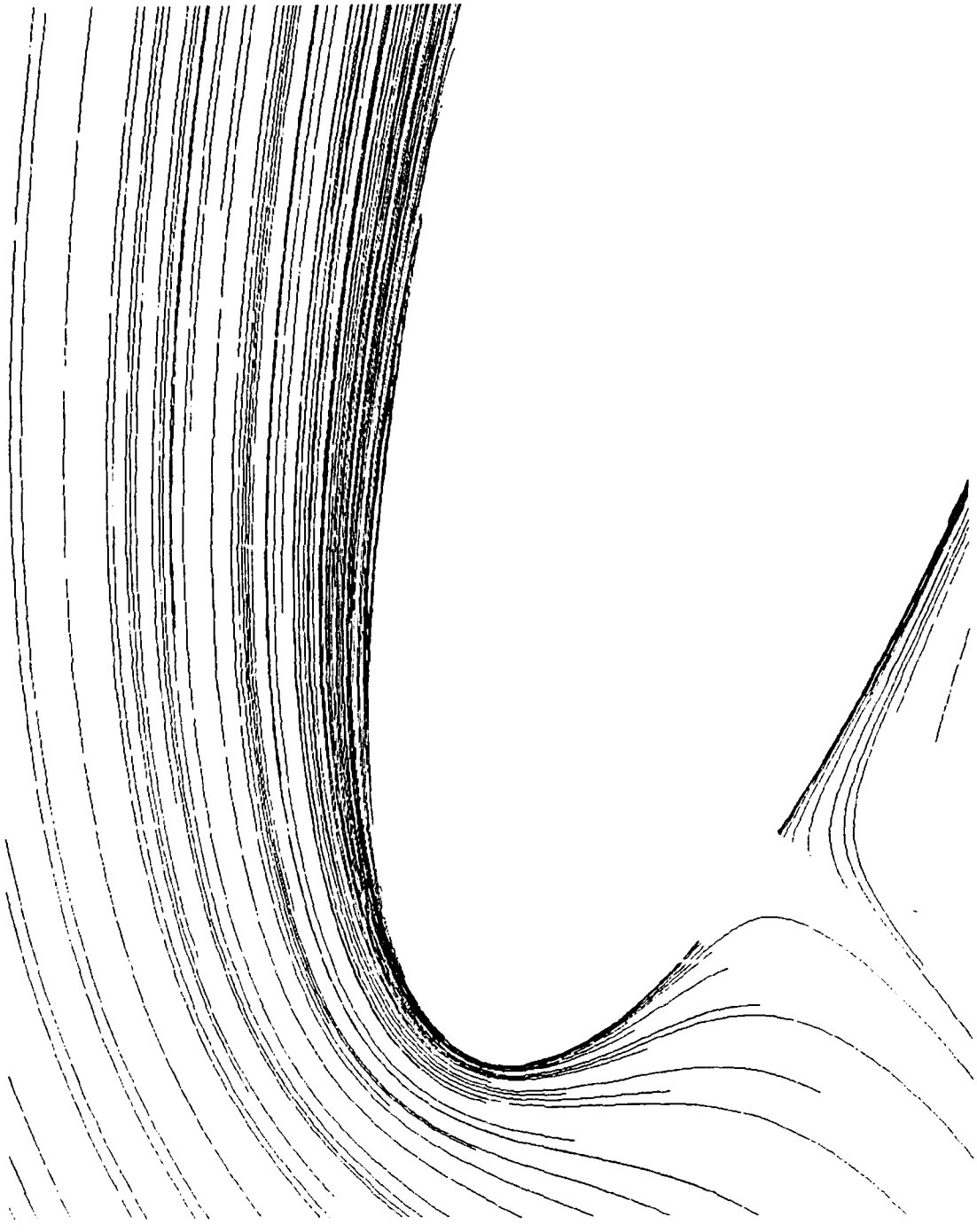

Figure 17. Leading Edge Instantaneous Streamlines, NACA 0012, M=0.4, k=0.01, α=17°

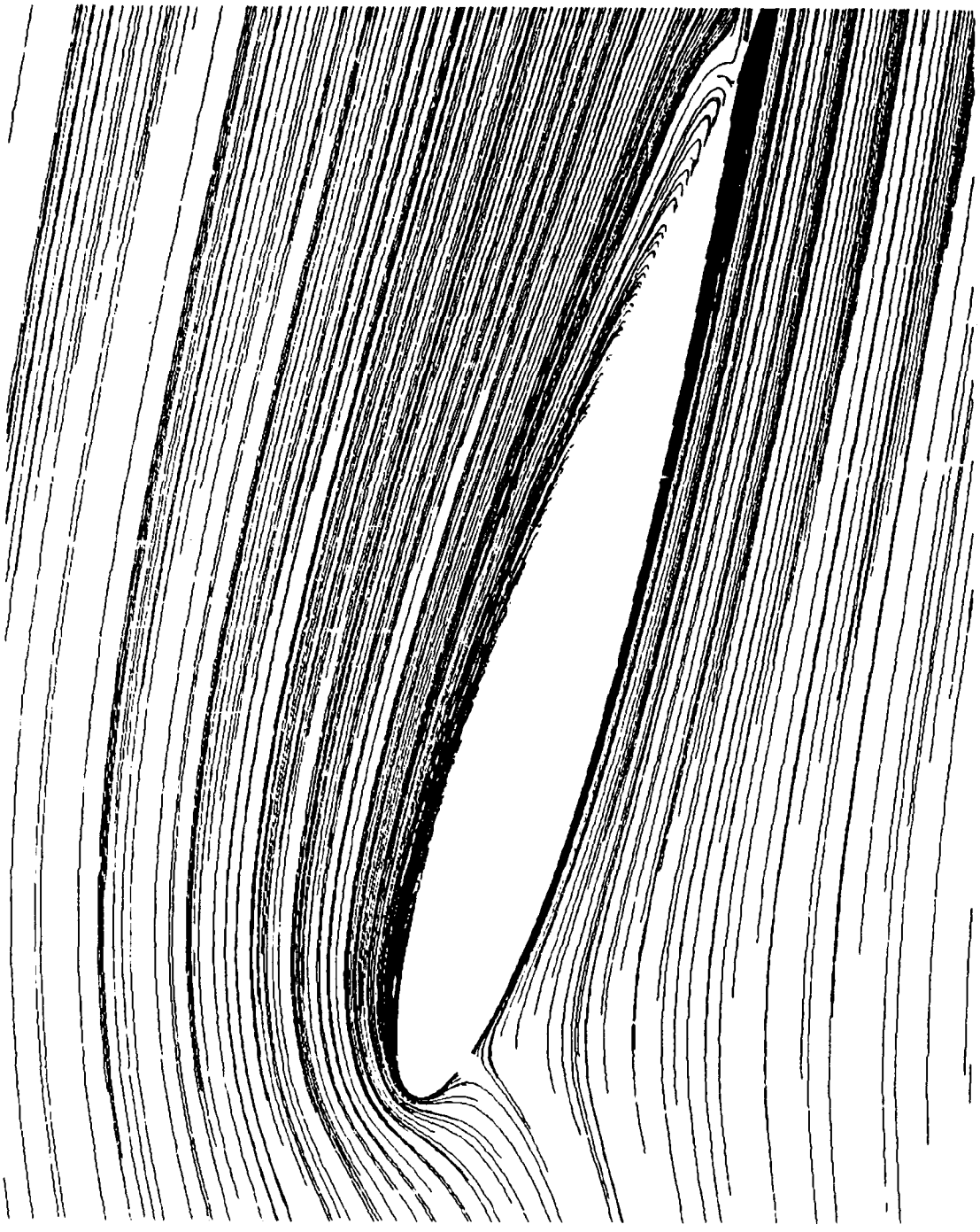

Figure 18. Instantaneous Streamlines, NACA 0012-63, M=0.4, k=0.01, α=17°

Figure 19. Leading Edge Instantaneous Streamlines, NACA 0012-63, M=0.4, k=0.01, α=17°

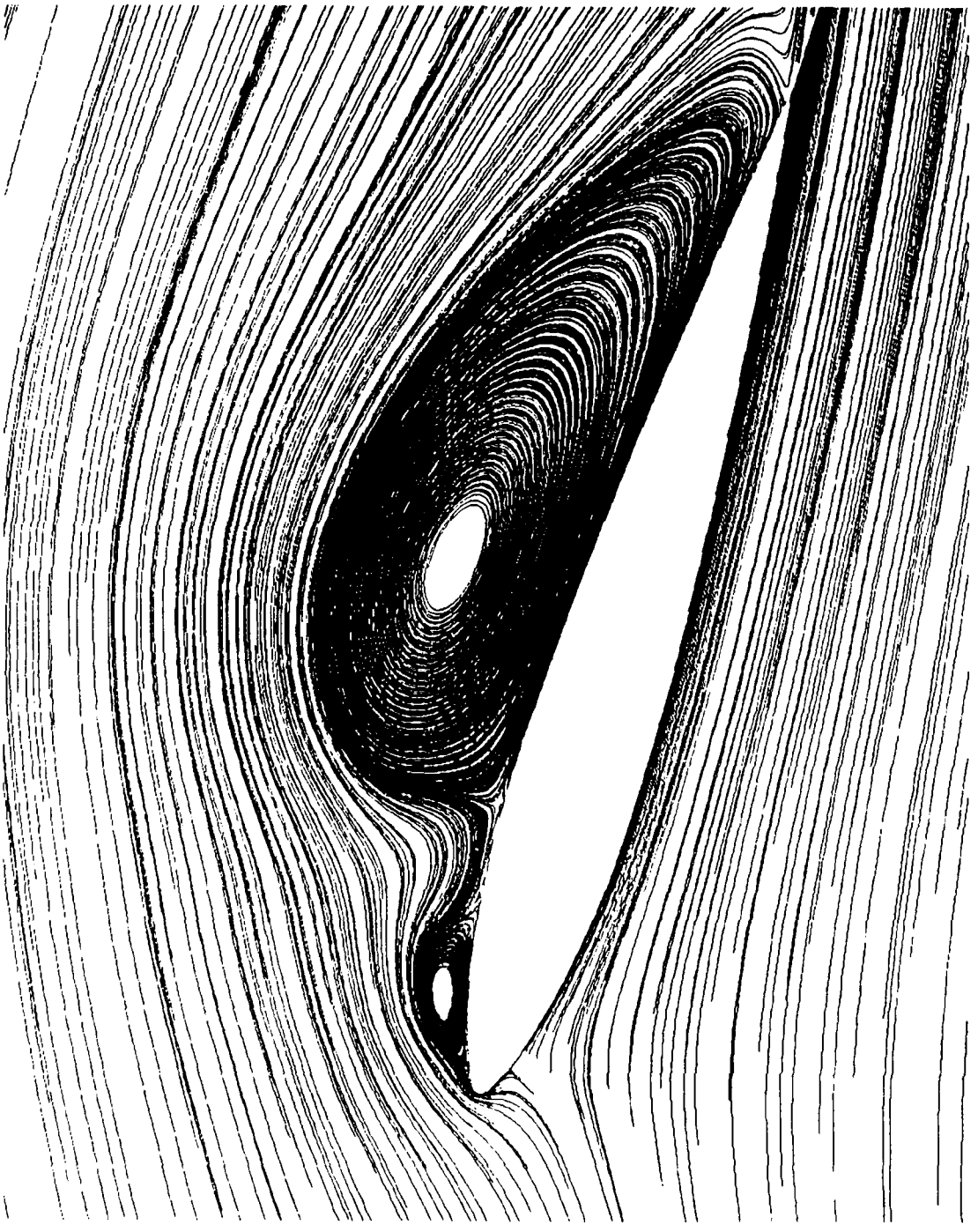

Figure 20. Instantaneous Streamlines, NACA 0012-33, M=0.4, k=0.01, α=17°

C. VORTEX FLOW DEVELOPMENT

1. Rapidly Pitching Airfoil

The development of the vortical flowfield as the angle of attack is increasing follows a consistent, general sequence of events. This sequence of events requires examination of the variation of several flow quantities as angle of attack is increased. Figures 21-33 illustrate the flow development stages using the velocity vector field, the pressure, and vorticity field contours for the NACA 0012-63 airfoil at M=0.4 and k=0.01. Figures 34-47 display the associated surface pressures and temperatures of the developing flow.

Initially, smooth streamlined flow is observed over the airfoil leading edge (Figures 21-22). As angle of attack increases beyond a certain critical limit depending on pitching rate, freestream Mach number, airfoil shape, and Reynolds number, small reverse flow regions develop on the upper surface aft of the suction peak and forward of the trailing edge due to the adverse pressure gradients encountered by the flow. Figure 23 shows the reversed boundary layer flow near the airfoil surface. Figure 24 displays the developing leading edge vortex.

The developing vortex forms on the upper surface just aft of the leading edge as a result of the combination of the accelerated flow near the suction peak and the reverse flow just aft of the suction peak. It is observed that the dynamic stall vortex initiates as a separated flow bubble, and rapidly

grows in size to form the characteristic leading edge dynamic stall vortical structure as the angle of attack increases. During this initiation process the primary vortex has a oval shape and becomes more rounded as the angle of attack increases. Figure 25 shows the oval dynamic stall vortex taking shape. Figure 26 displays the relative strength of the developing vortex.

The vortex grows in size and strengthens, becoming approximately circular as angle of attack increases, and moves away from the airfoil. Figure 27 shows the development of the vortex as it moves downstream. Passage of the dynamic stall vortex over the airfoil surface induces reverse velocities and significantly contributes to the development of reversed flows. A secondary vortex originates near the leading edge and also grows as angle of attack increases. Figures 28 and 29 show the development of a secondary vortex as the primary vortex moves downstream and promotes reverse flow over the whole upper airfoil surface.

At the trailing edge, flow separation is initiated at approximately the same time as at the leading edge, and a pressure gradient exists between the lower and upper surfaces. The upper surface reverse flow combines with the high pressure flow from the lower surface to form a counter-clockwise vortex at the trailing edge shortly before dynamic stall occurs. Figures 30 and 31 show the developed trailing edge vortex 0.6° angle of attack before peak lift coefficient is obtained. As

the primary vortex grows and moves downstream past approximately 60-70% chord, dynamic stall occurs, usually accompanied by development of a small, tertiary vortex at the leading edge. Figures 32 and 33 display the position of the dynamic stall vortex 0.4° angle of attack after attainment of peak lift coefficient.

By this time, the primary vortex is centered at a distance greater than the airfoil maximum thickness from the airfoil and combines with lower surface high pressure flow to energize the counter-clockwise vortex at the trailing edge.

With further increase in angle of attack, the airfoil enters the deep stall regime, and the freestream flow has no effect on the upper surface flow characteristics. In this case, the aerodynamic behavior of the airfoil is greatly determined by the vortical flow field formed by the shed vortices, and the strength of the vortices themselves determine the flow over the upper surface. Figures 48 and 49 show the NACA 0012-33 airfoil in deep stall at 0.3 Mach, k=0.02, α=22.0° and demonstrate that the vortices can combine to form a large vortical region and entrain other secondary flows. In this case a counterclockwise vortex was formed between the paired primary and secondary vortices and the airfoil.

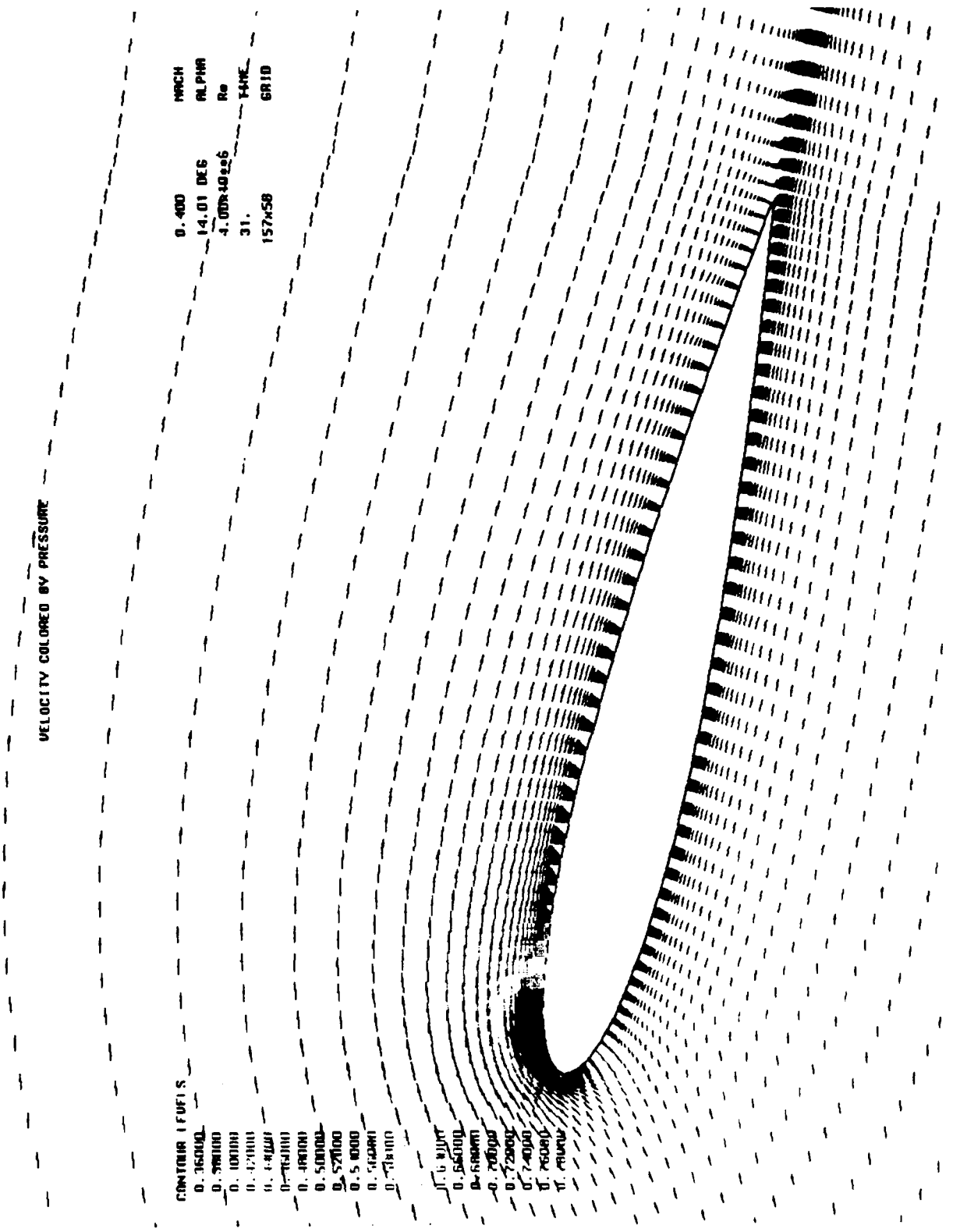

Figure 21. Velocity Field, NACA 0012-63, M=0.4, k=0.01, α=14.0°

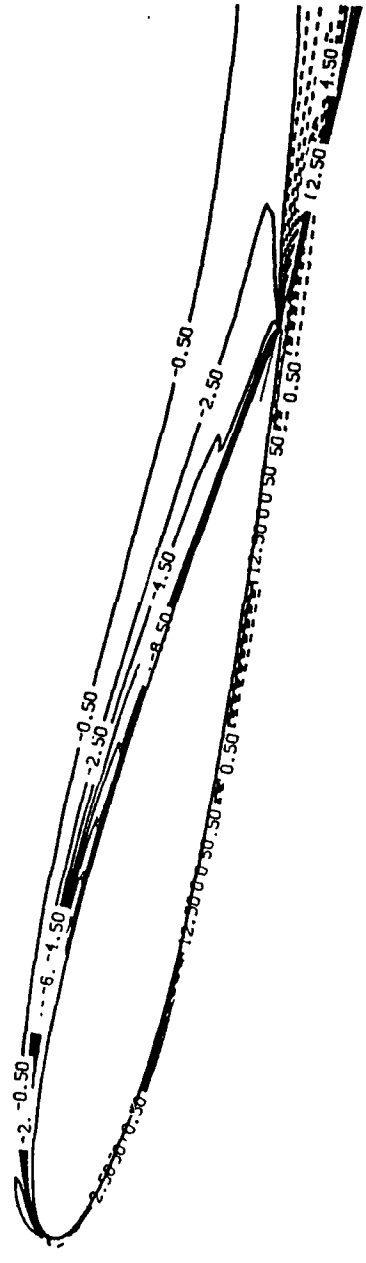

Figure 22. Vorticity Contours, NACA 0012-63, M=0.3, k=0.01, α=14.0°

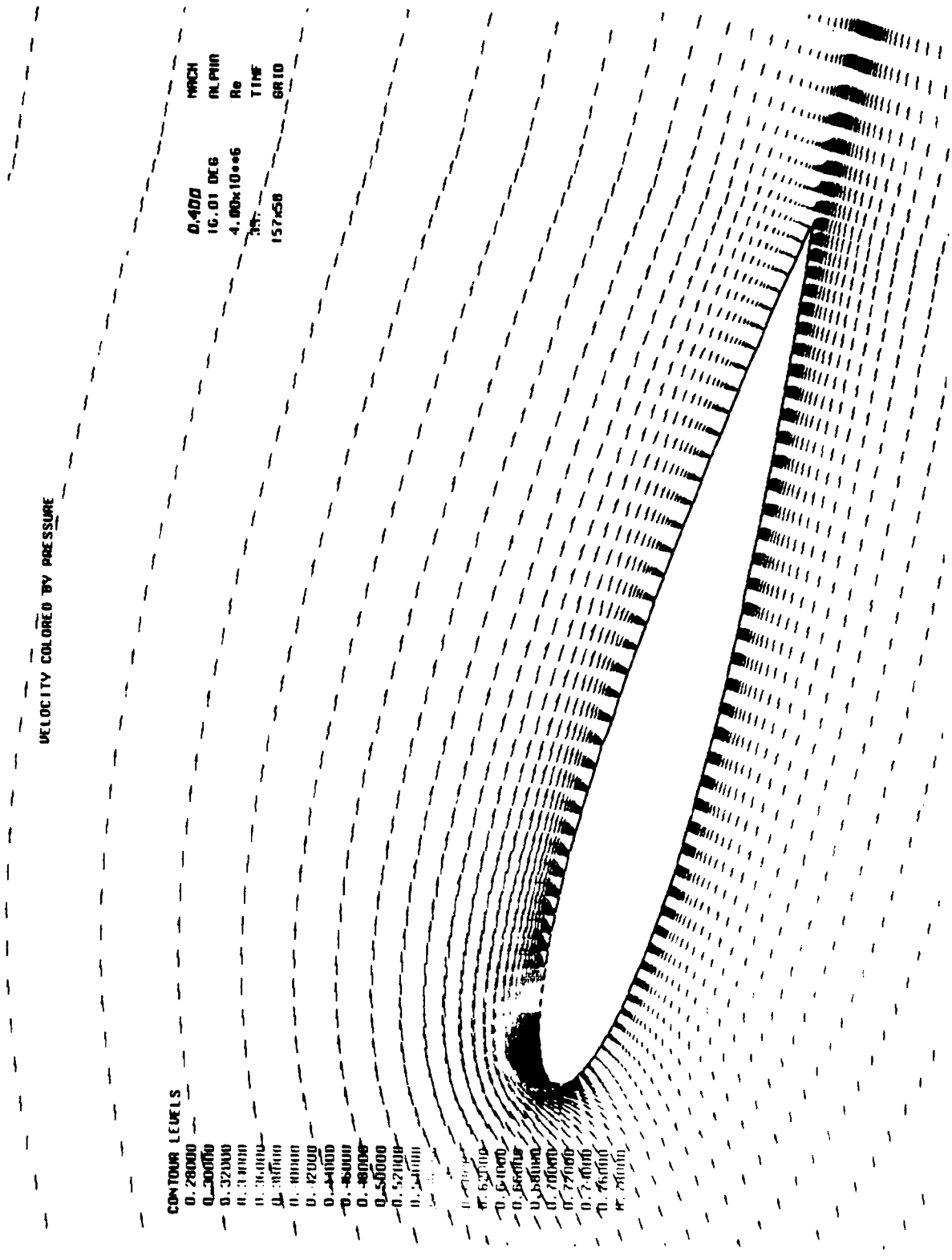

Figure 23. Velocity Field, NACA 0012-63, M=0.4, k=0.01, α=16.0°

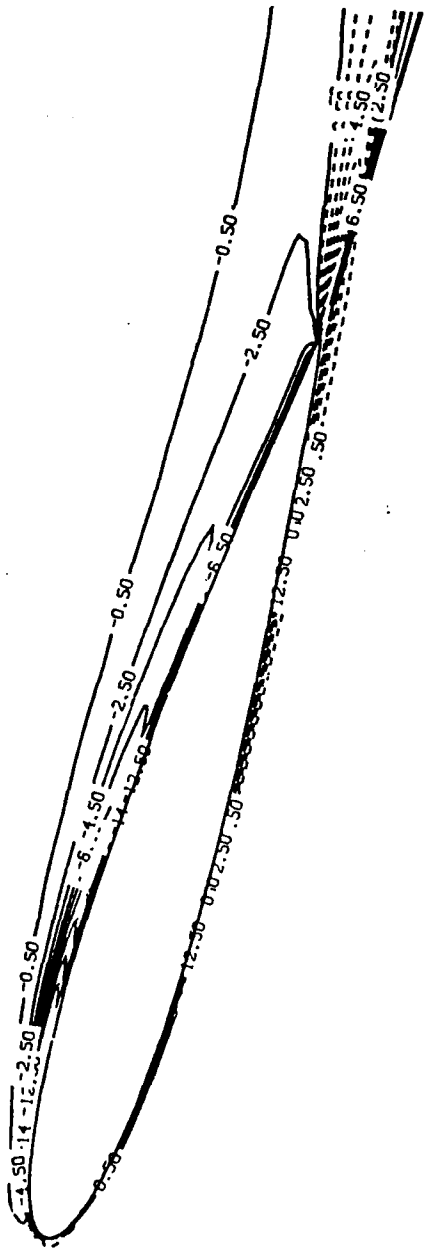

Figure 24. Vorticity Contours, NACA 0012-63, M=0.4, k=0.01, α=16.0°

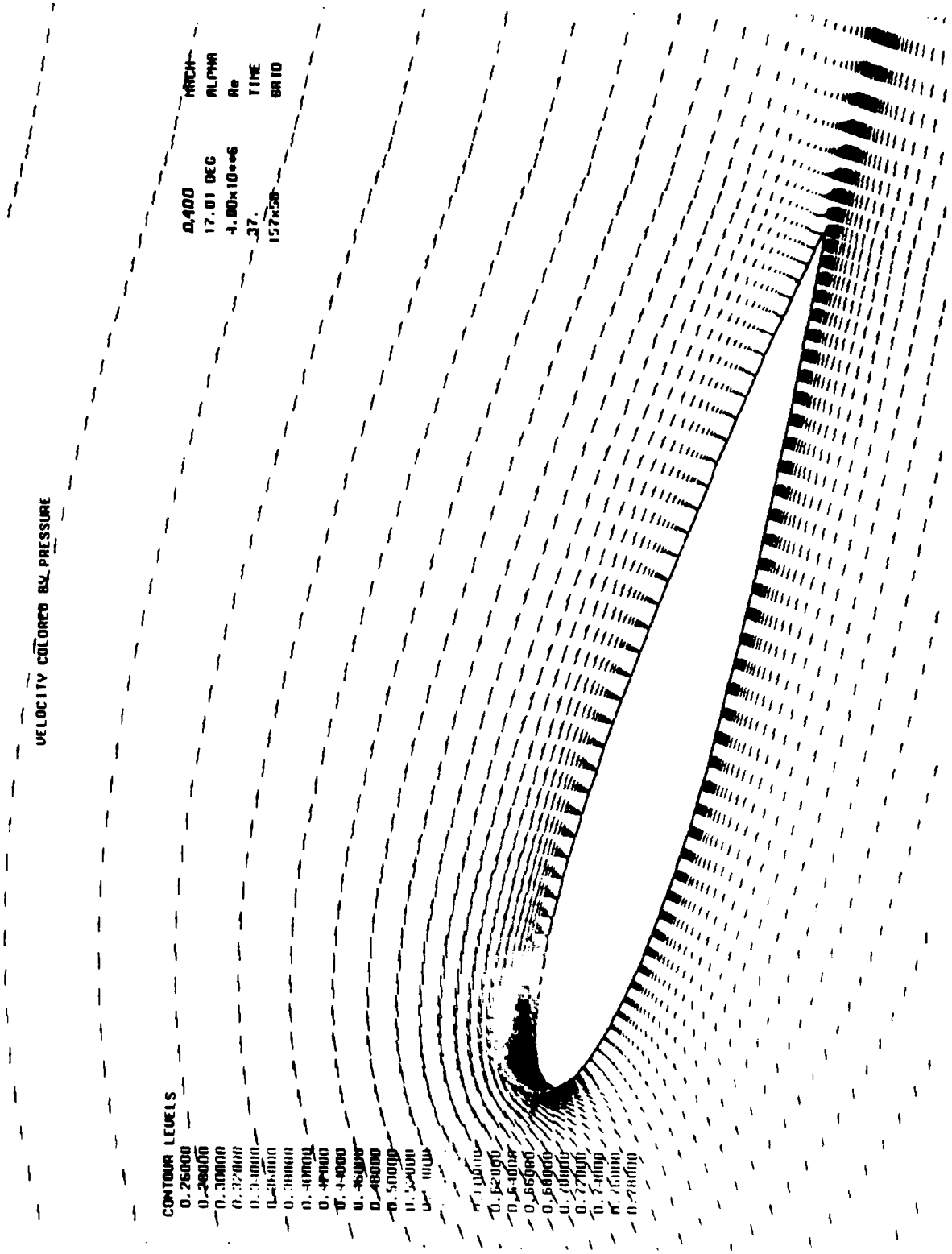

Figure 25. Velocity Field, NACA 0012-63, M=0.4, k=0.01, α=17.0°

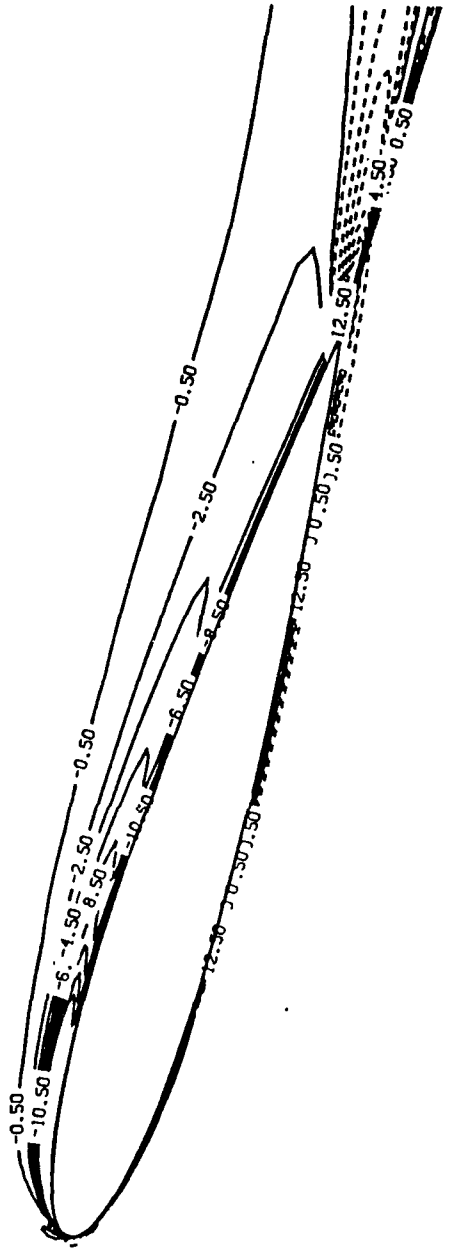

Figure 26. Vorticity Contours, NACA 0012-63, M=0.3, k=0.01, α=17.0°

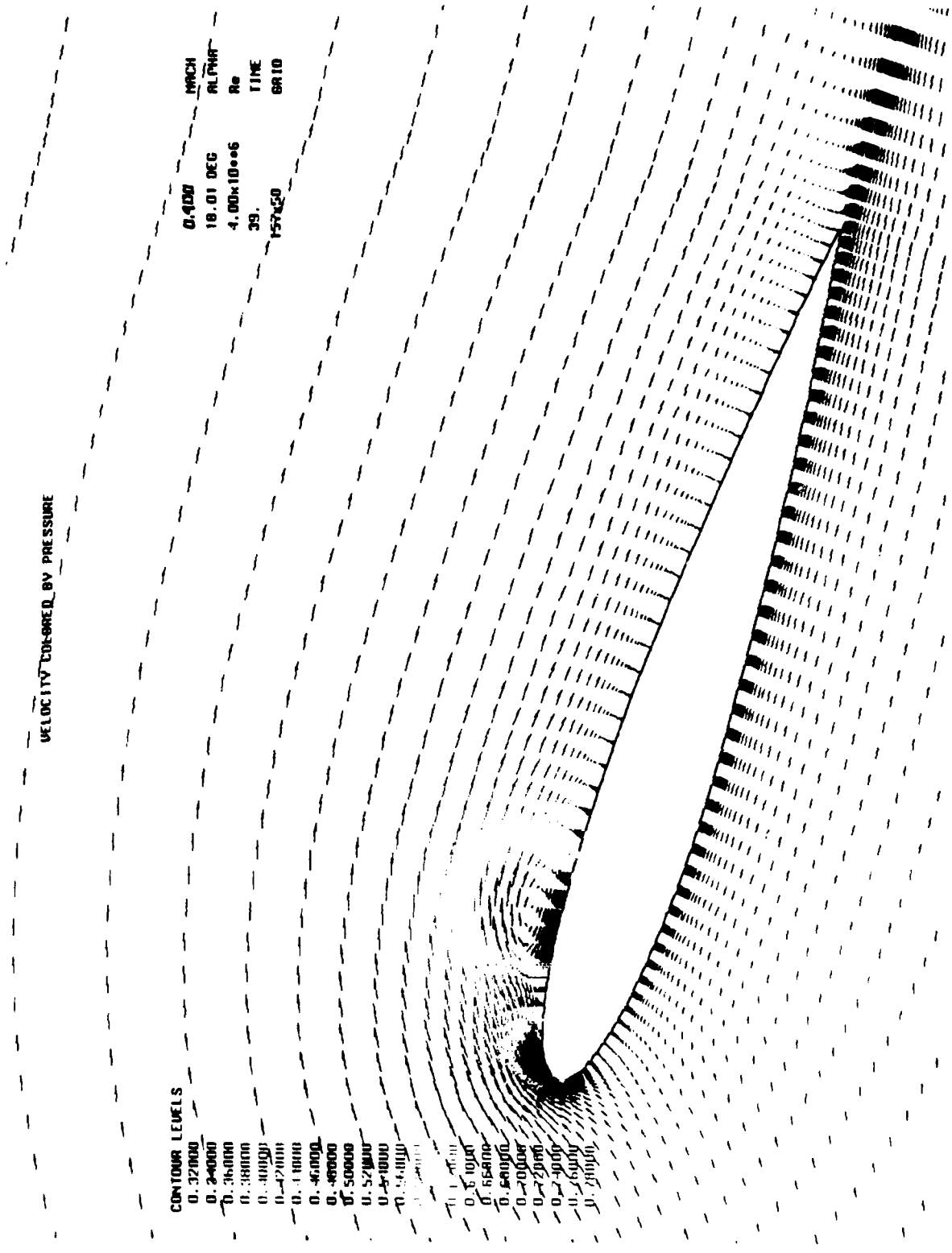

Figure 27. Velocity Field, NACA 0012-63, M=0.4, k=0.01, α=18.0°

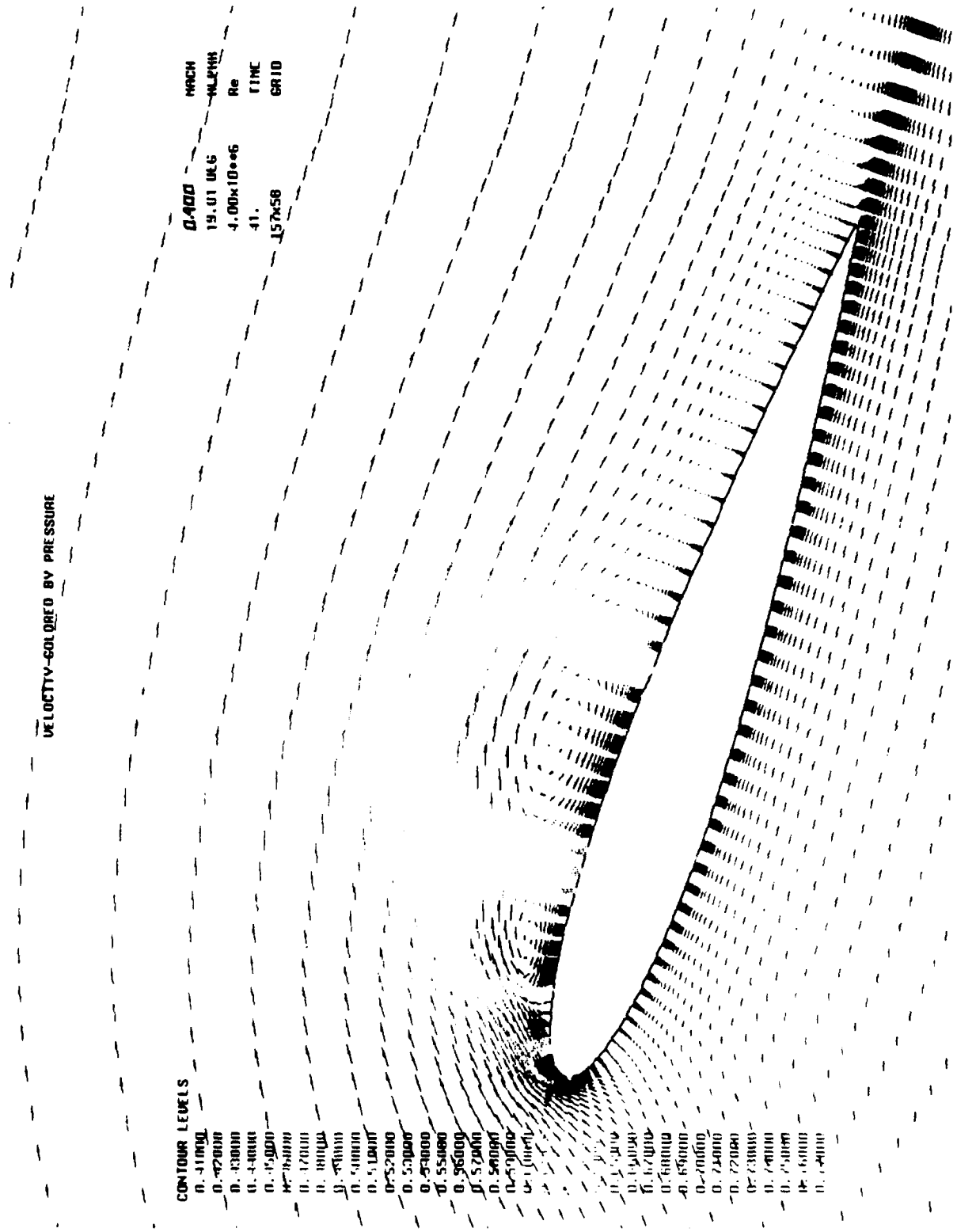

Figure 28. Velocity Field, NACA 0012-63, M=0.4. k=0.01, α=19.0°

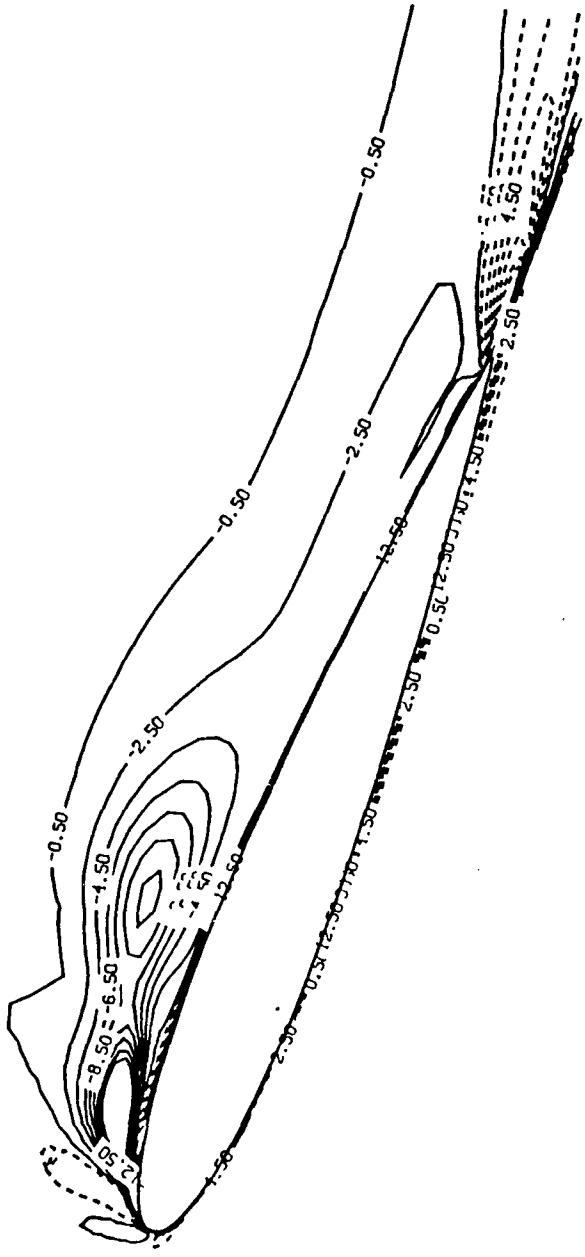

Figure 29. Vorticity Contours, NACA 0012-63, M=0.4, k=0.01, α=19.0°

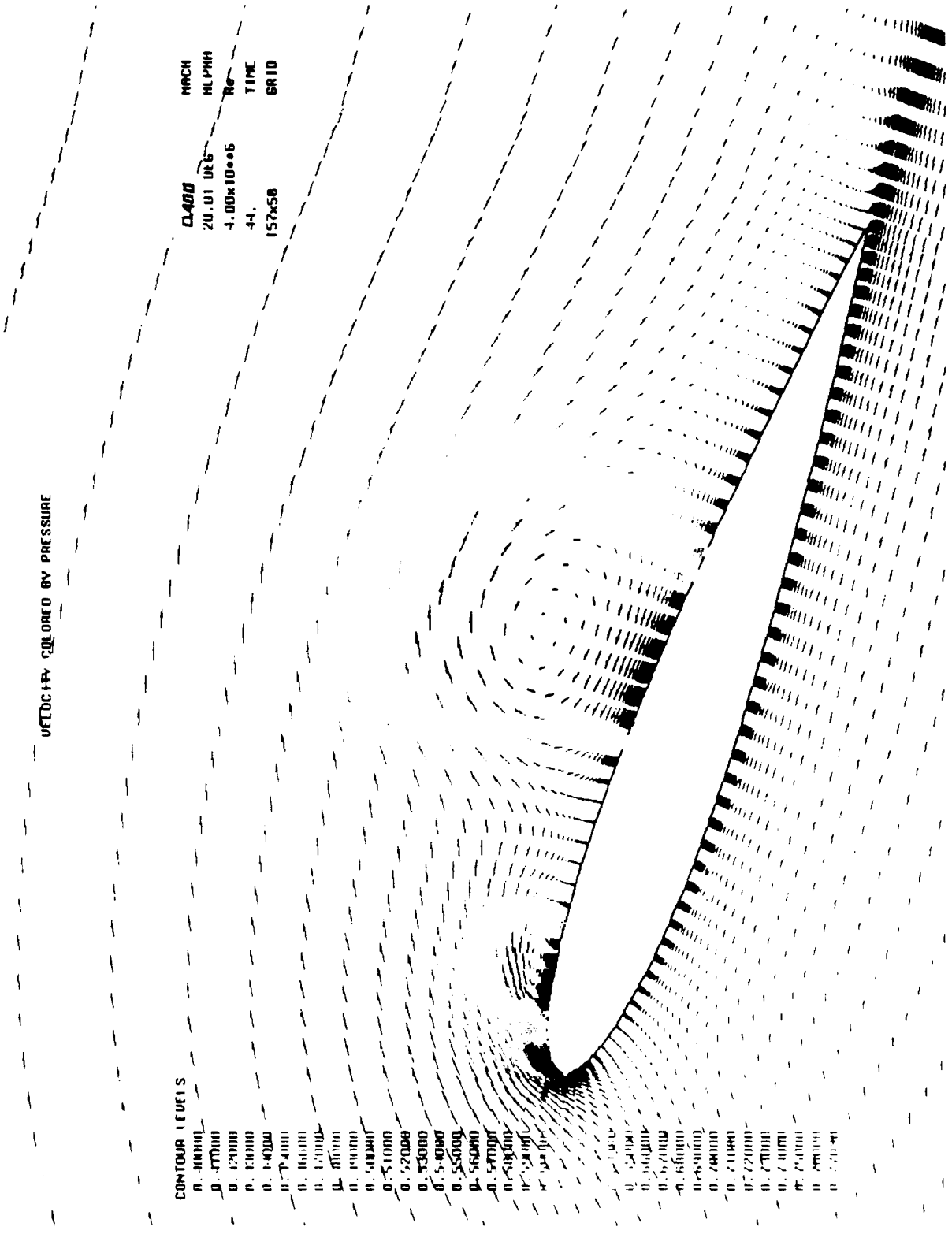

Figure 30. Velocity Field, NACA 0012-63, M=0.4, k=0.01, α=20.0°

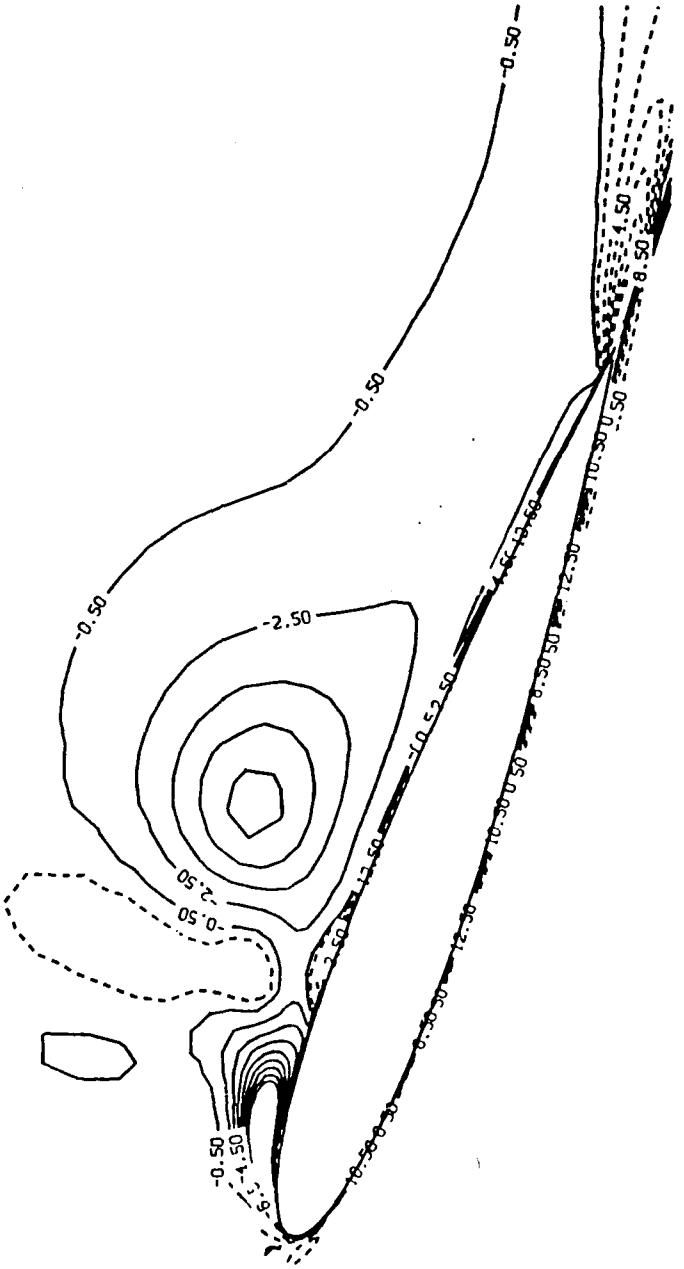

Figure 31. Vorticity Contours, NACA 0012-63, M=0.3, k=0.01, α=20.0°

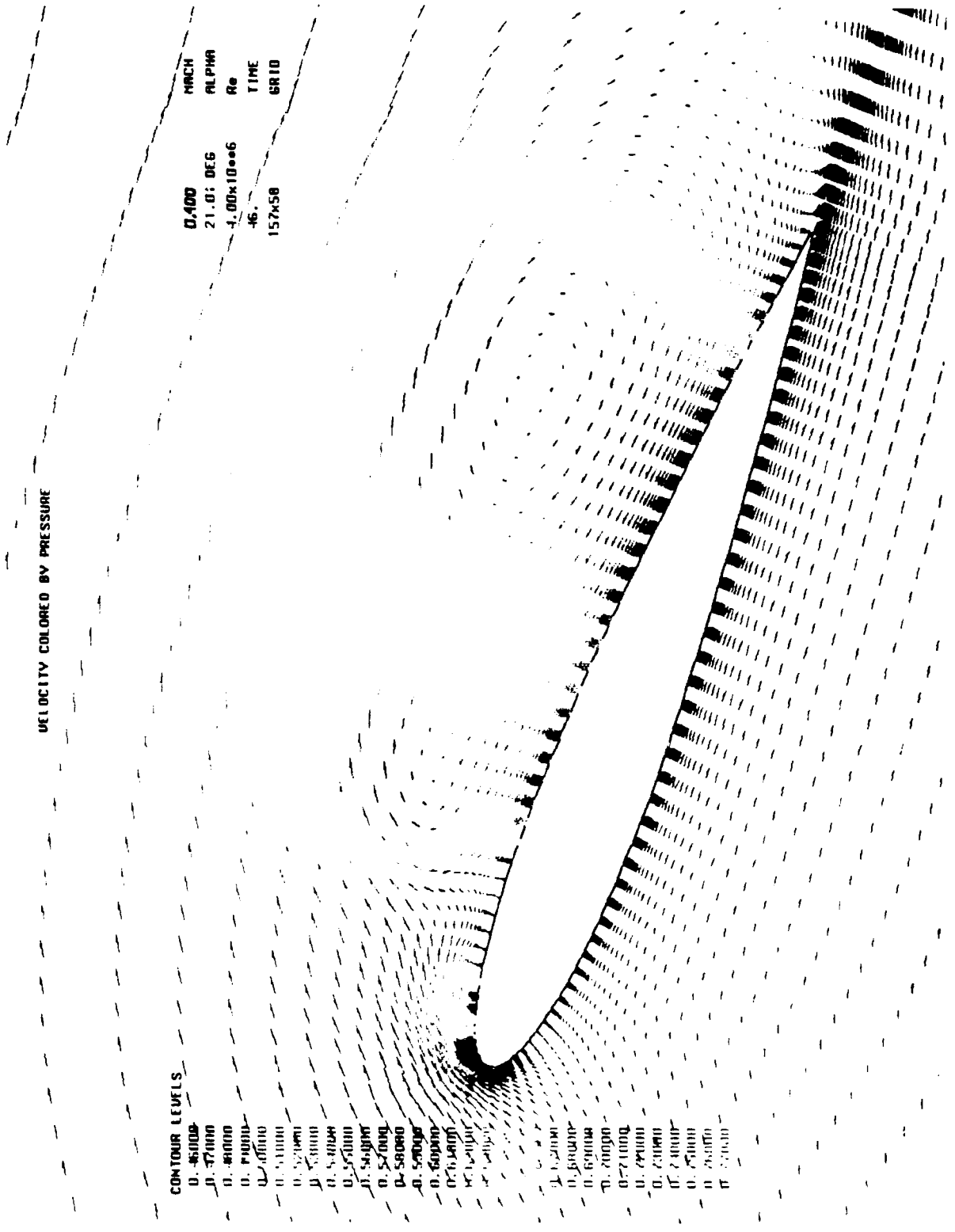

Figure 32. Velocity Field, NACA 0012-63, M=0.4, k=0.01, α=21.0°

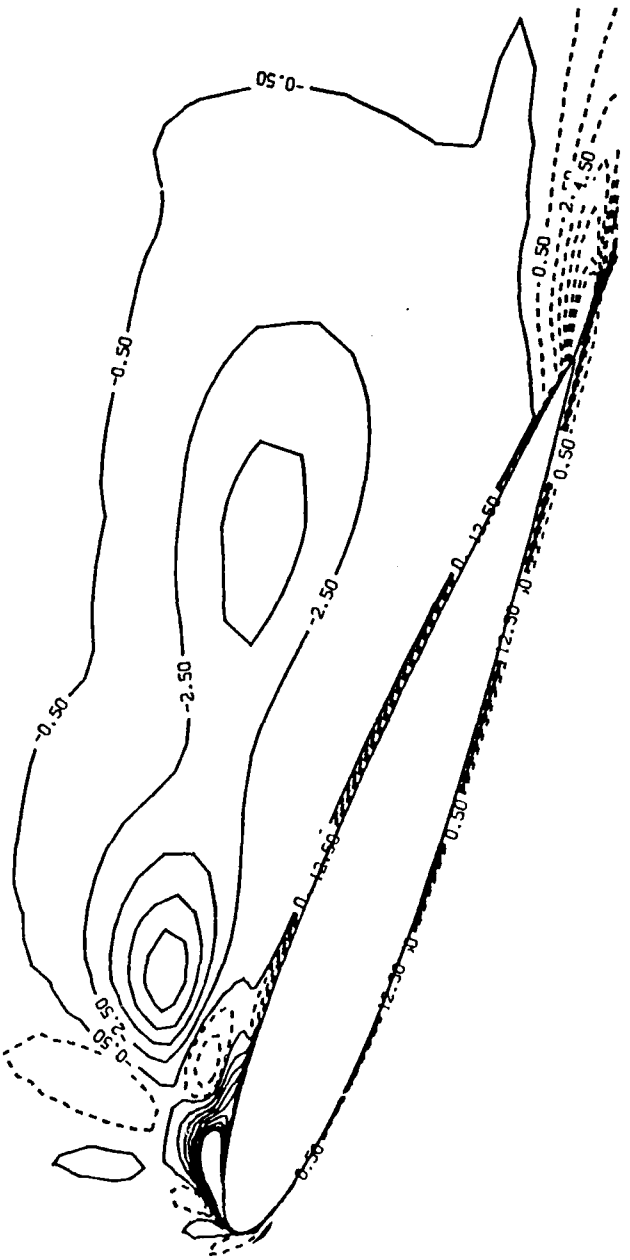

Figure 33. Vorticity Contours, NACA 0012-63, M=0.4, k=0.01, α=21.0°

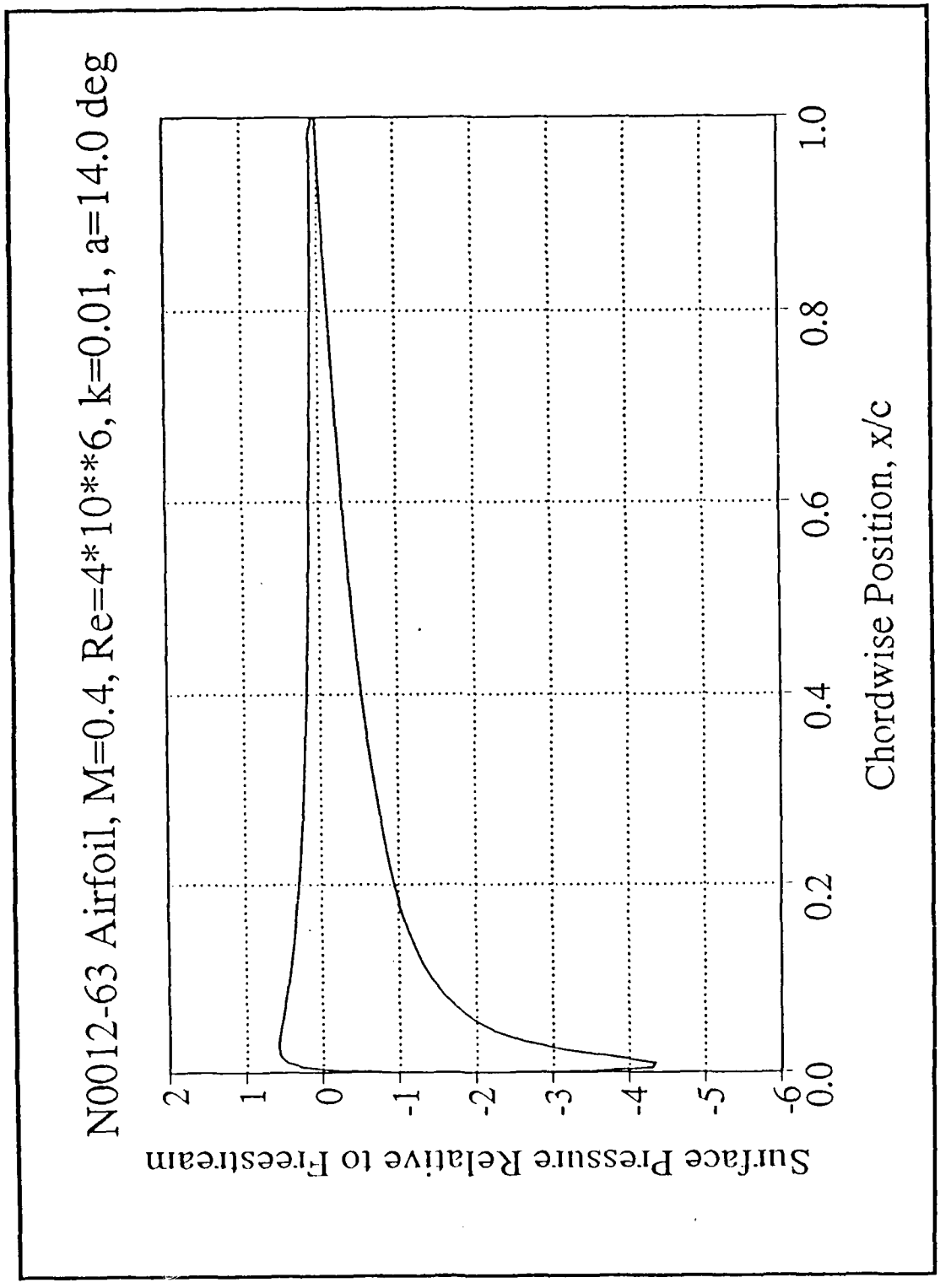

Figure 34. Surface Pressure, NACA 0012-63, M=0.4, k=0.01, α=14.0°

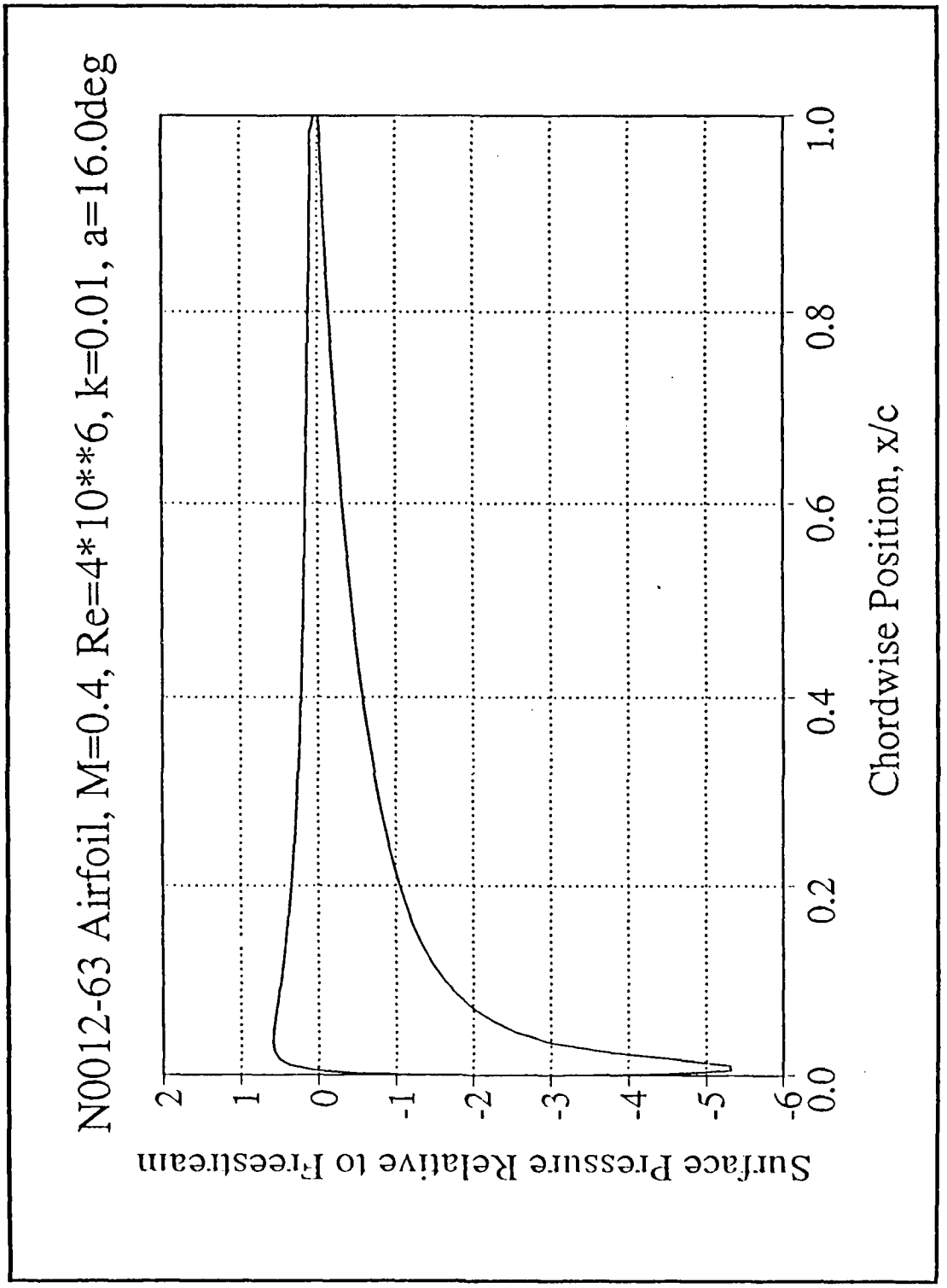

Figure 35. Surface Pressure, NACA 0012-63, M=0.4, k=0.01, α=16.0°

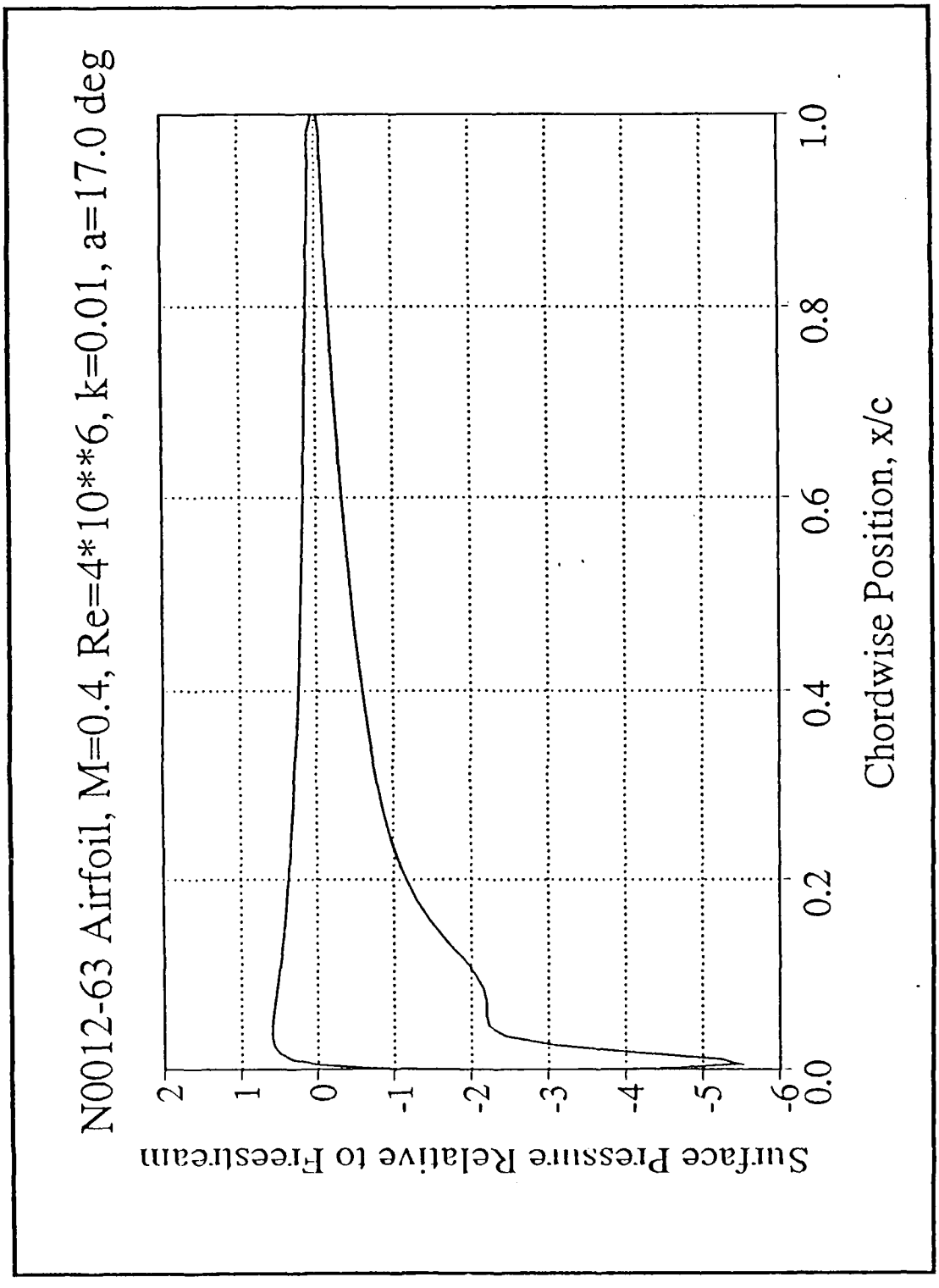

Figure 36. Surface Pressure, NACA 0012-63, M=0.4, k=0.01, α=17.0°

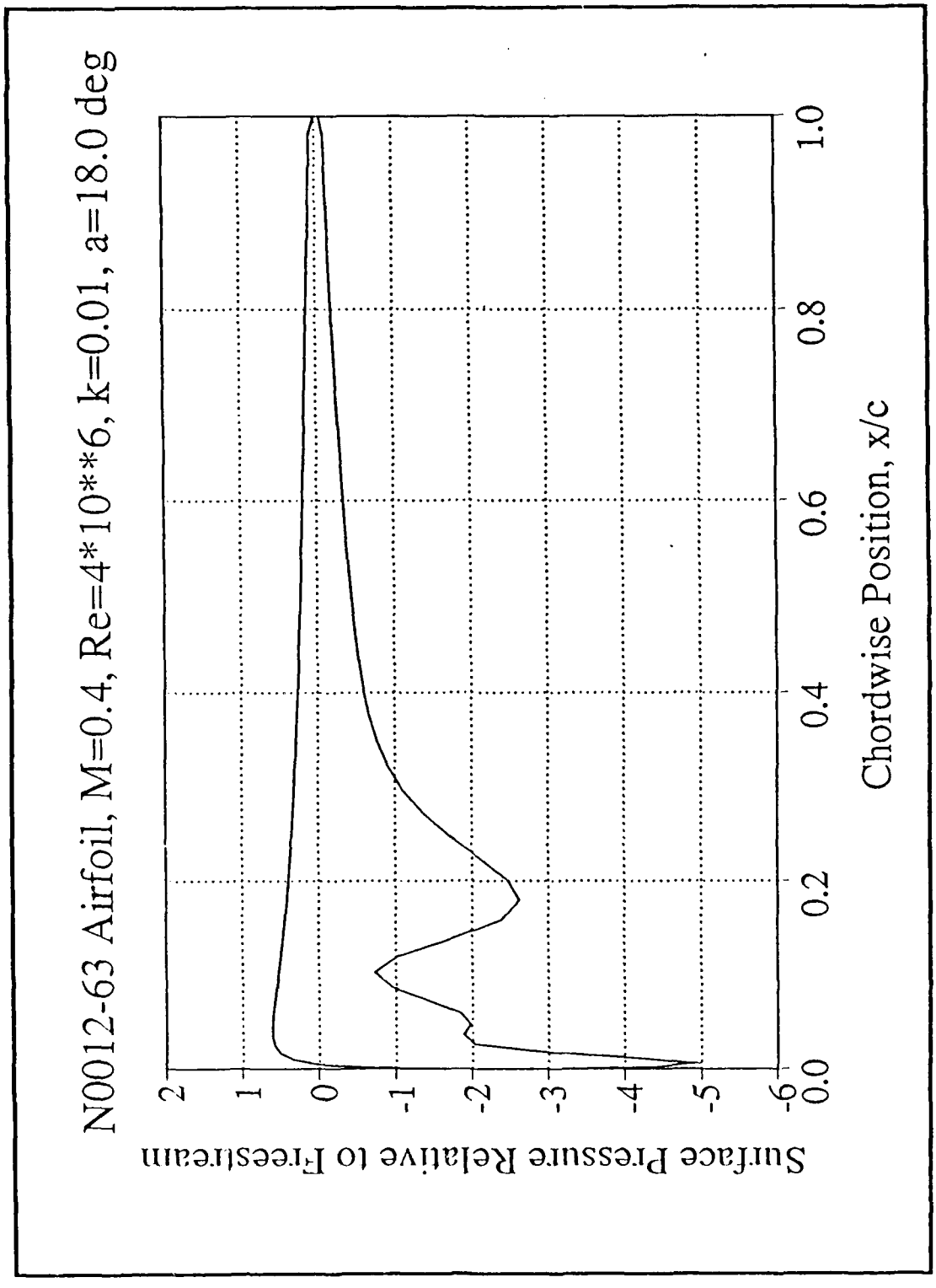

Figure 37. Surface Pressure, NACA 0012-63, M=0.4, k=0.01, α=18.0°

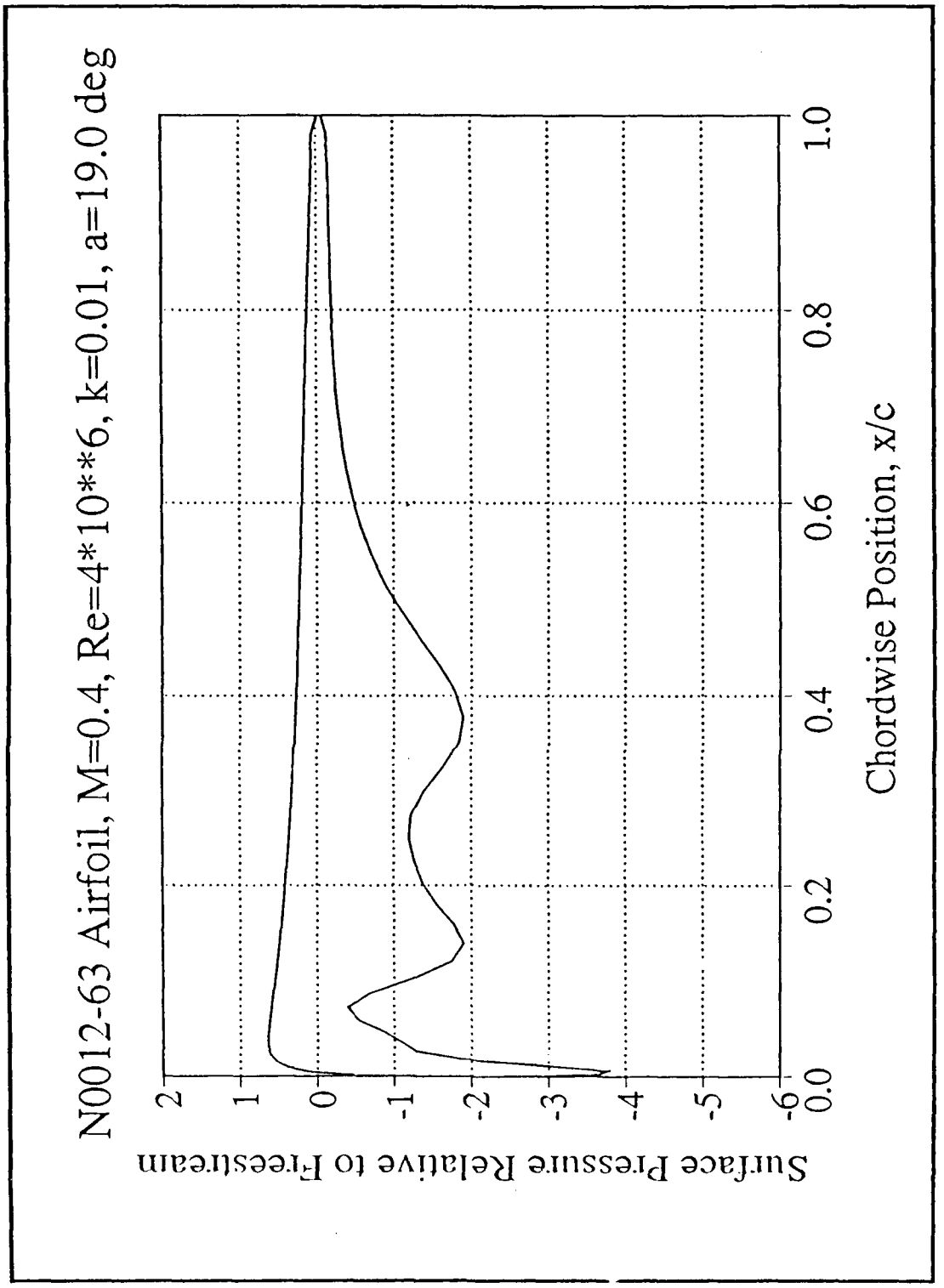

Figure 38. Surface Pressure, NACA 0012-63, M=0.4, k=0.01, α=19.0°

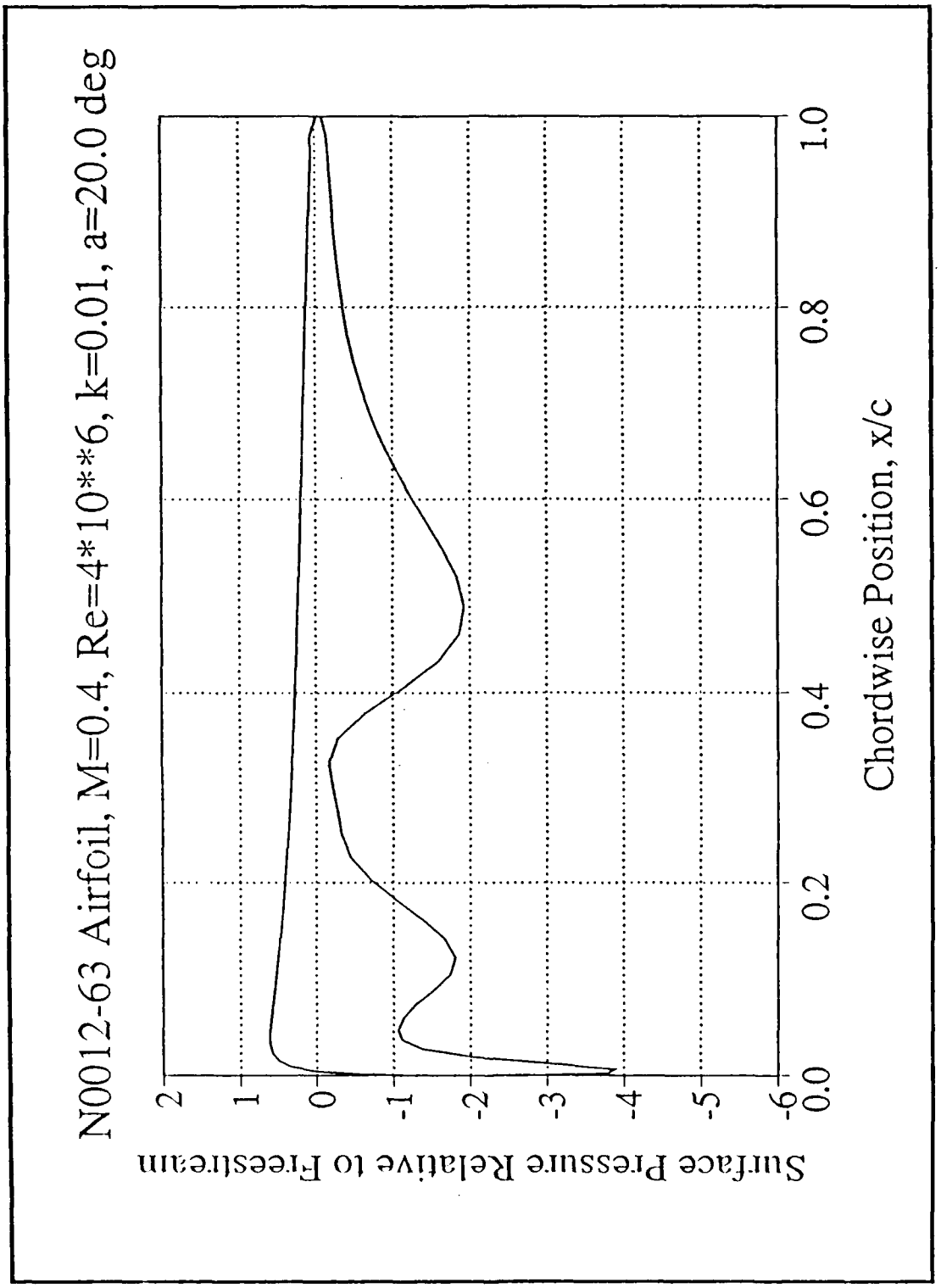

Figure 39. Surface Pressure, NACA 0012-63, M=0.4, k=0.01, α=20.0°

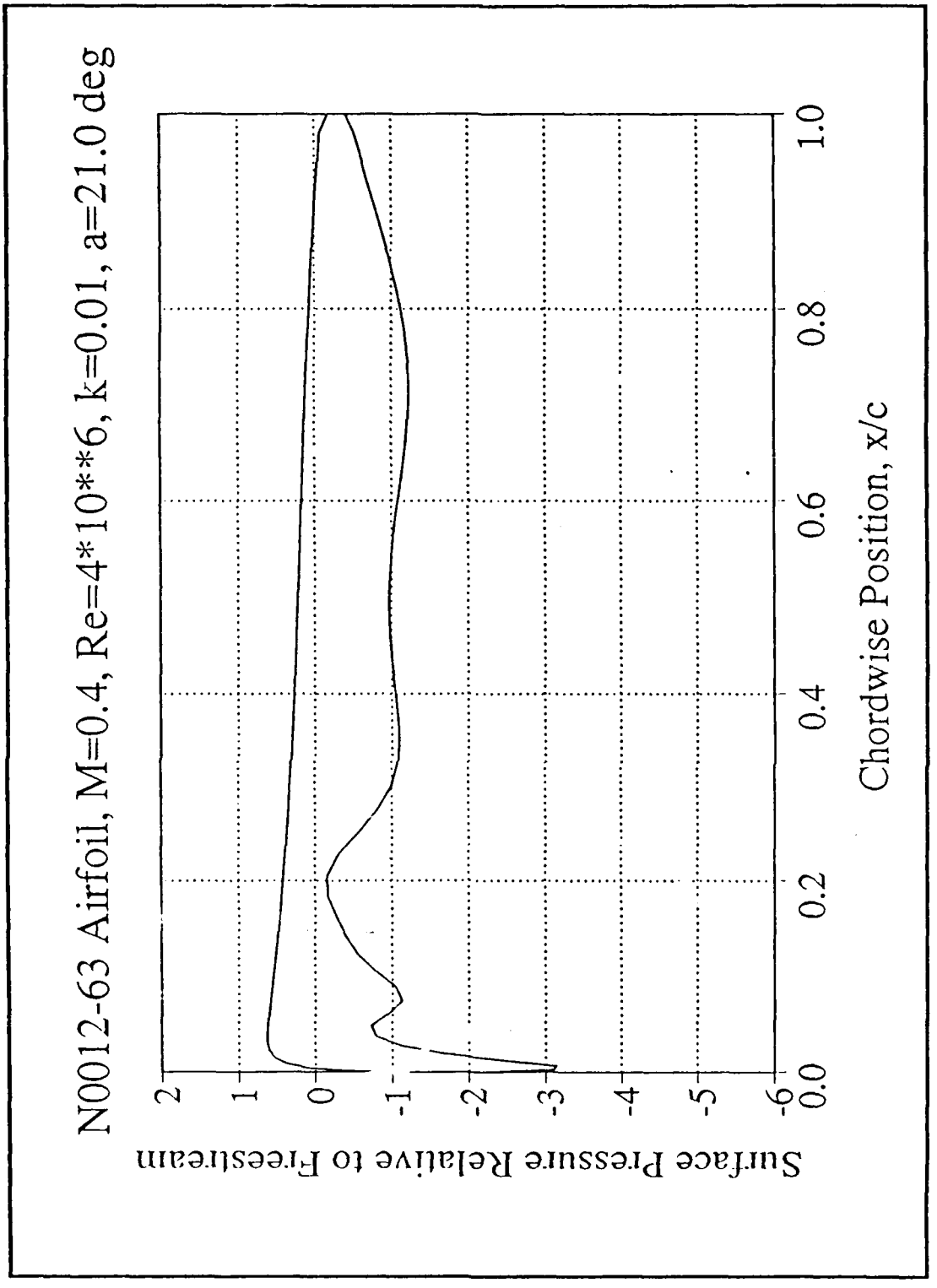

Figure 40. Surface Pressure, NACA 0012-63, M=0.4, k=0.01, α=21.0°

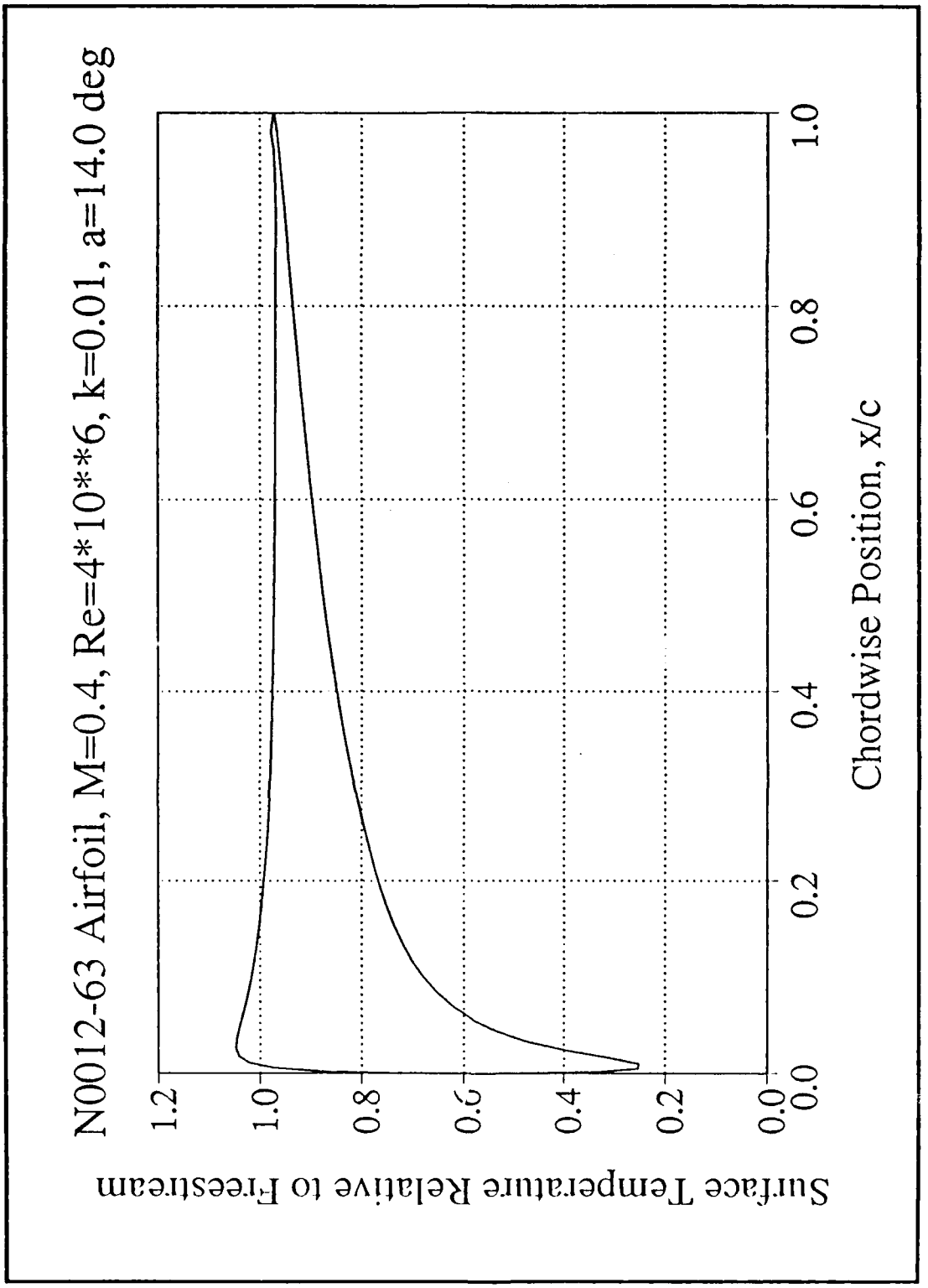

Figure 41. Surface Temperature, NACA 0012-63, M=0.4, k=0.01, α=14.0°

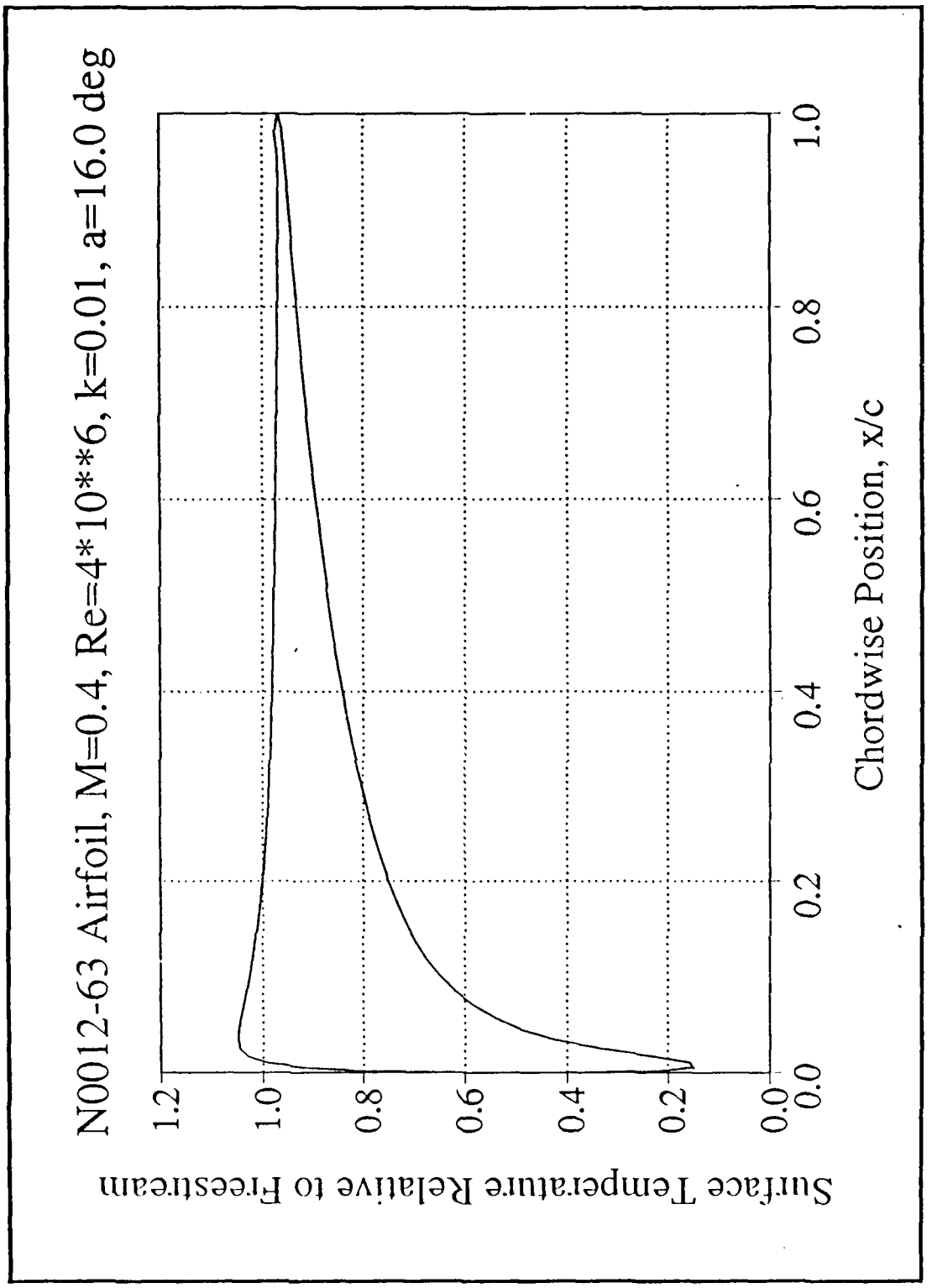

Figure 42. Surface Temperature, NACA 0012-63, M=0.4, k=0.01, α=16.0°

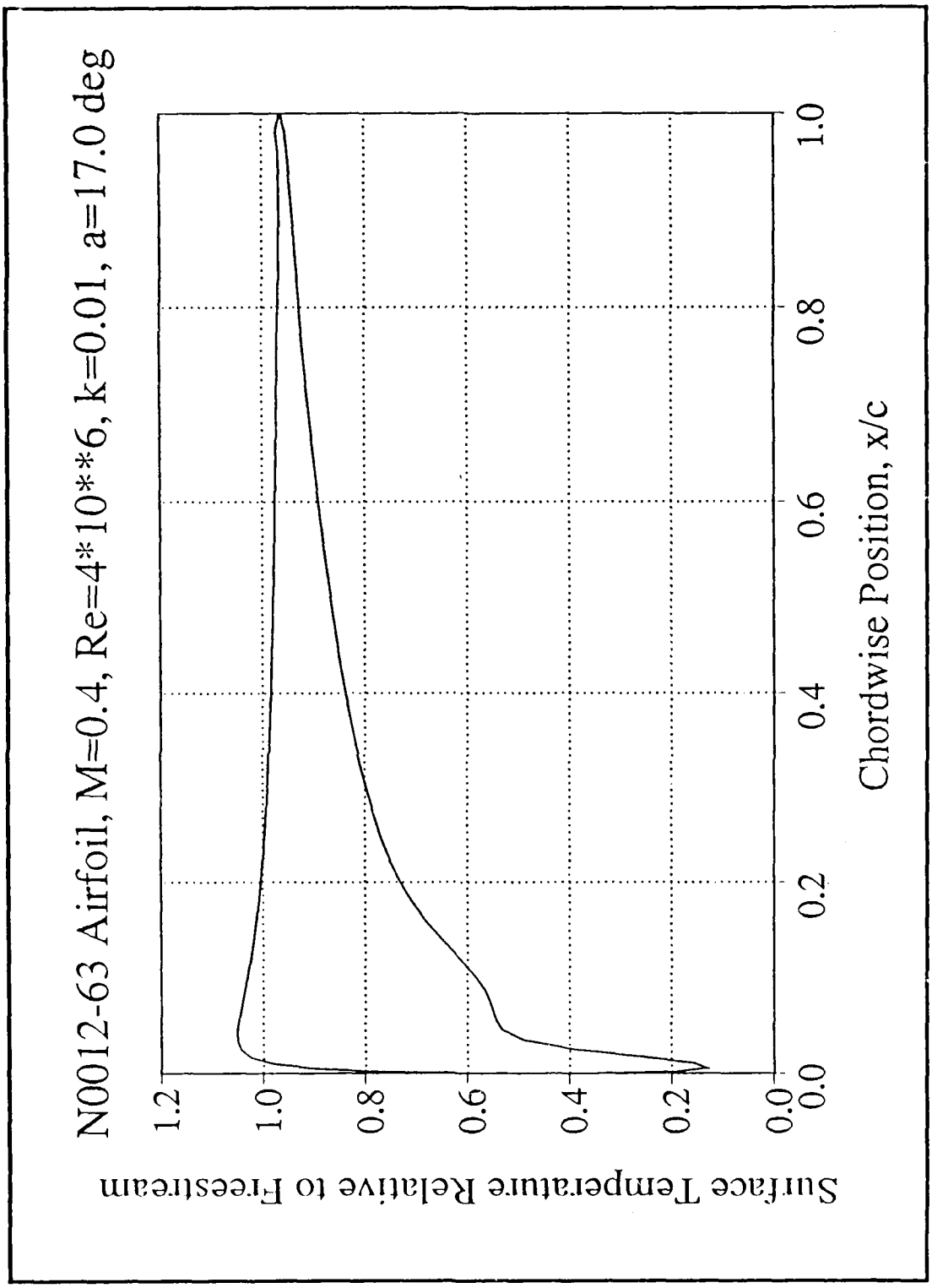

Figure 43. Surface Temperature, NACA 0012-63, M=0.4, k=0.01, α=17.0°

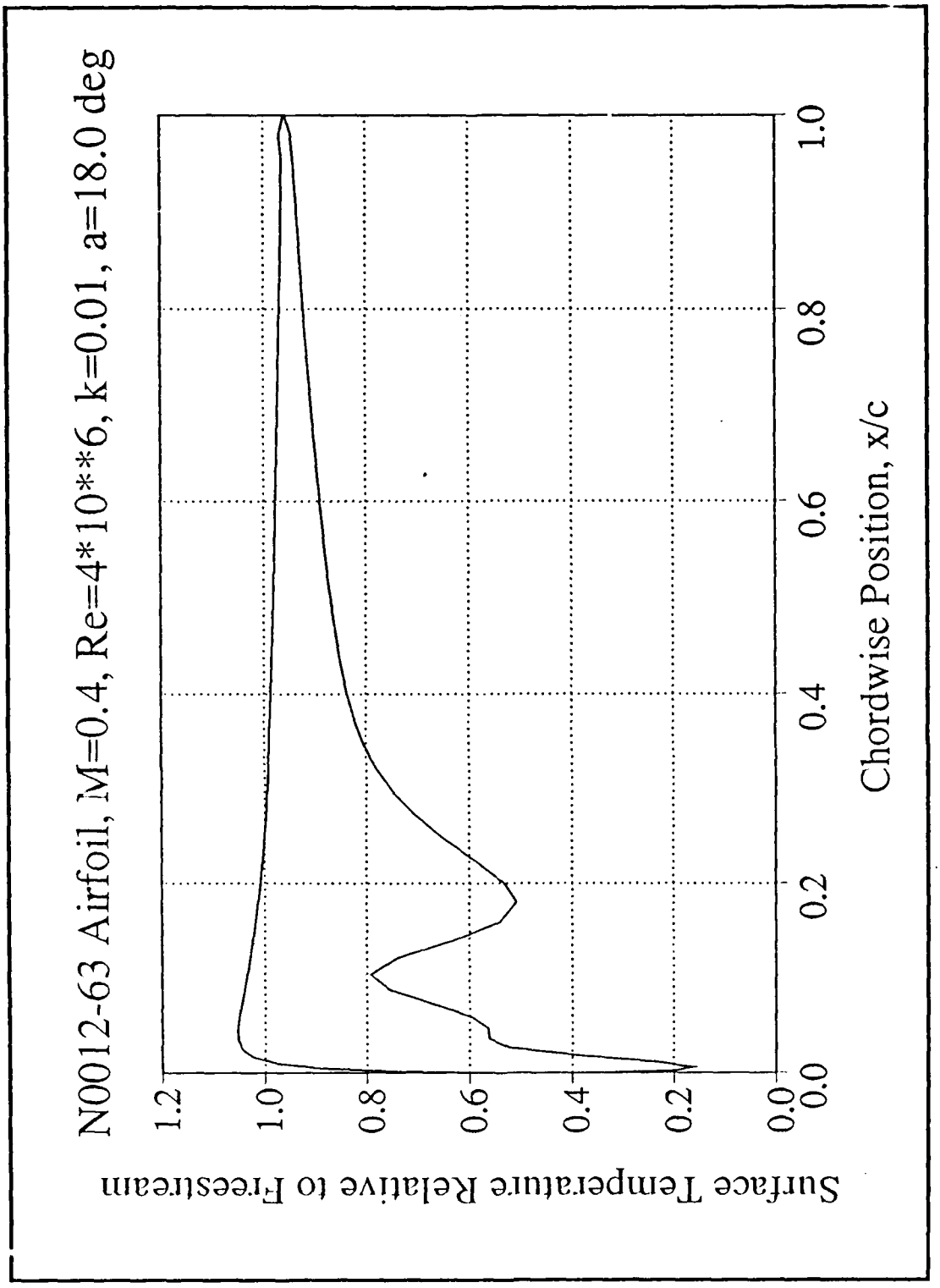

Figure 44. Surface Temperature, NACA 0012-63, M=0.4, k=0.01, α=18.0°

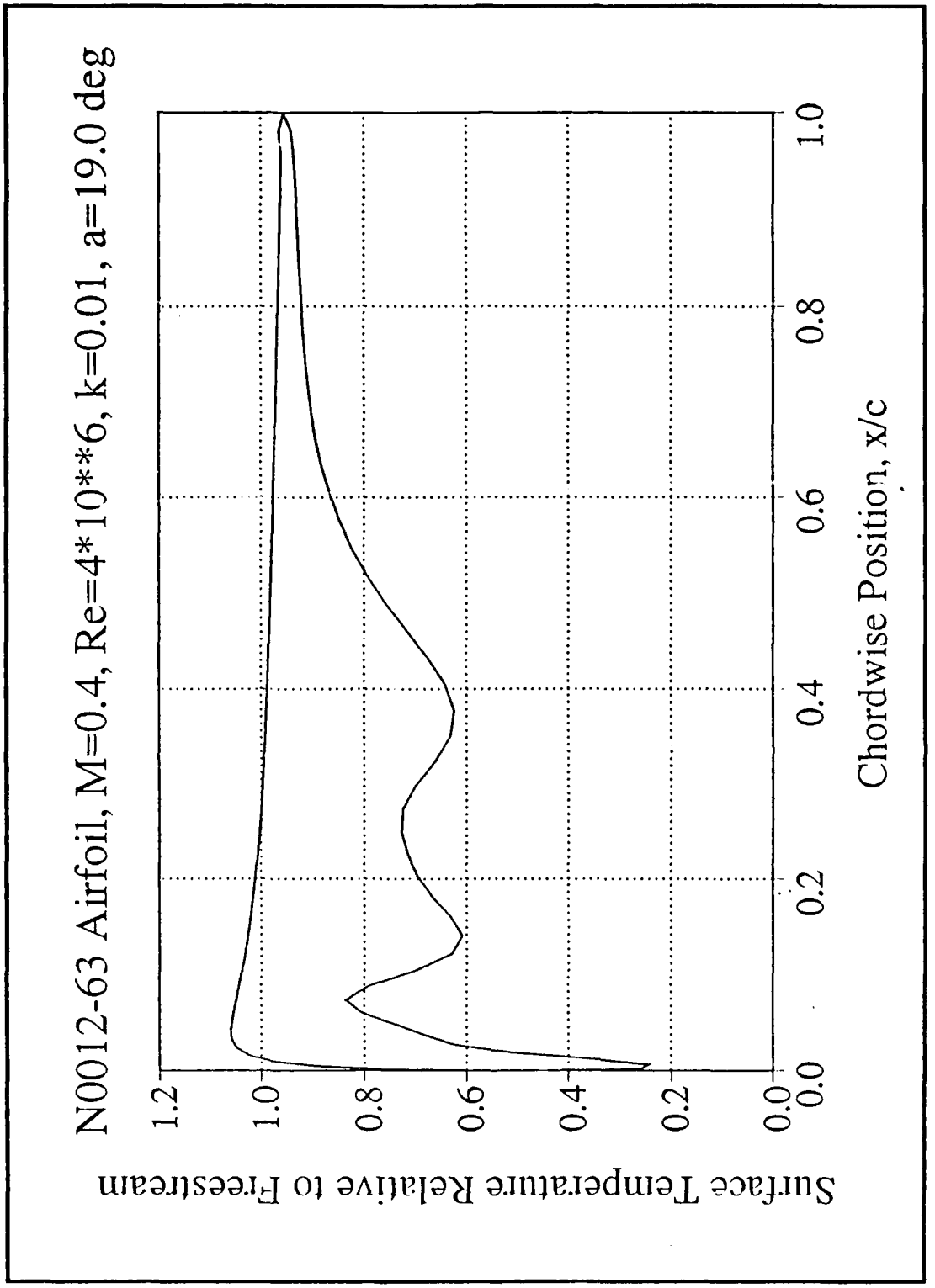

Figure 45. Surface Temperature, NACA 0012-63, M=0.4, k=0.01, α=19.0°

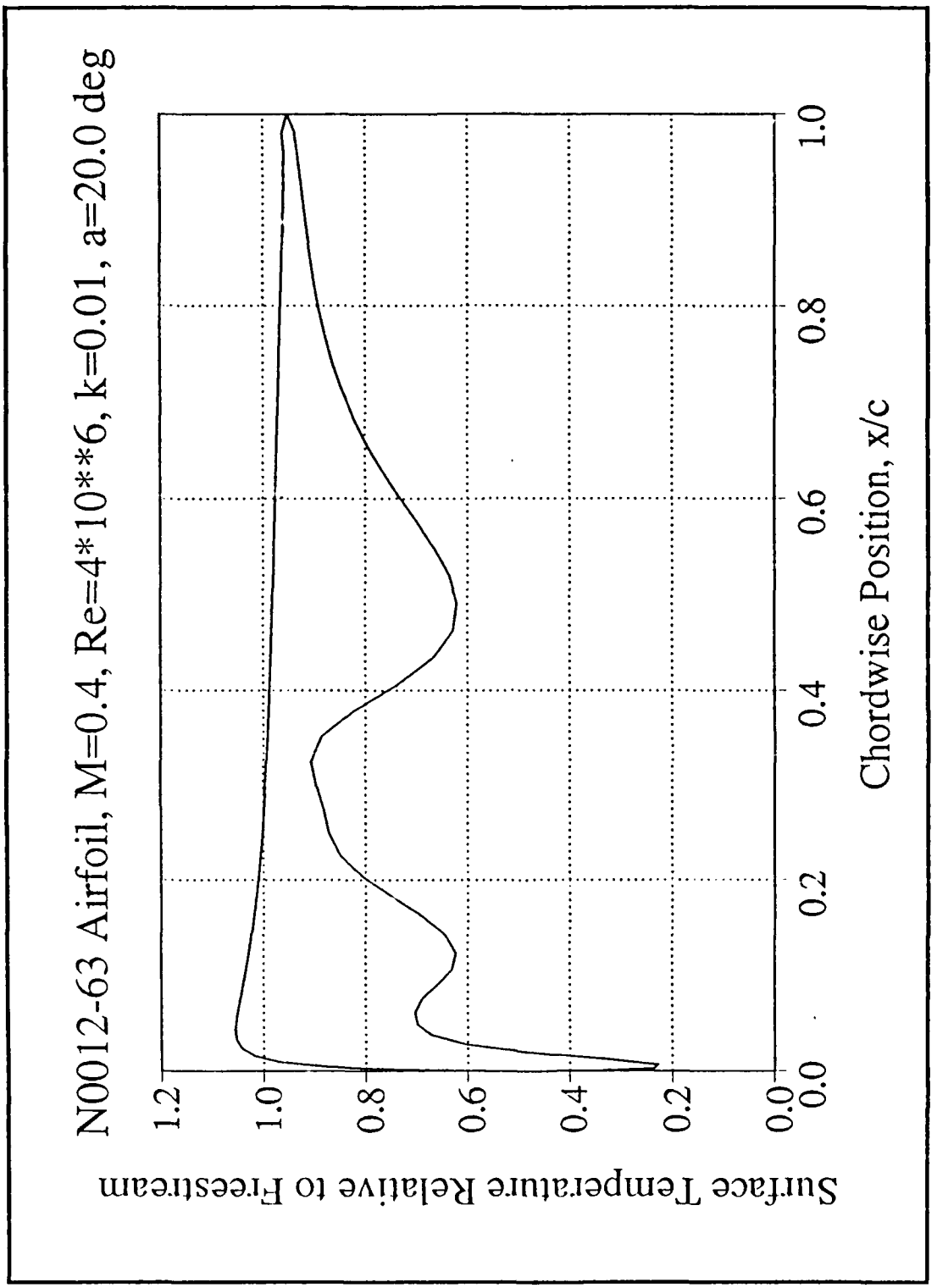

Figure 46. Surface Temperature, NACA 0012-63, M=0.4, k=0.01, α=20.0°

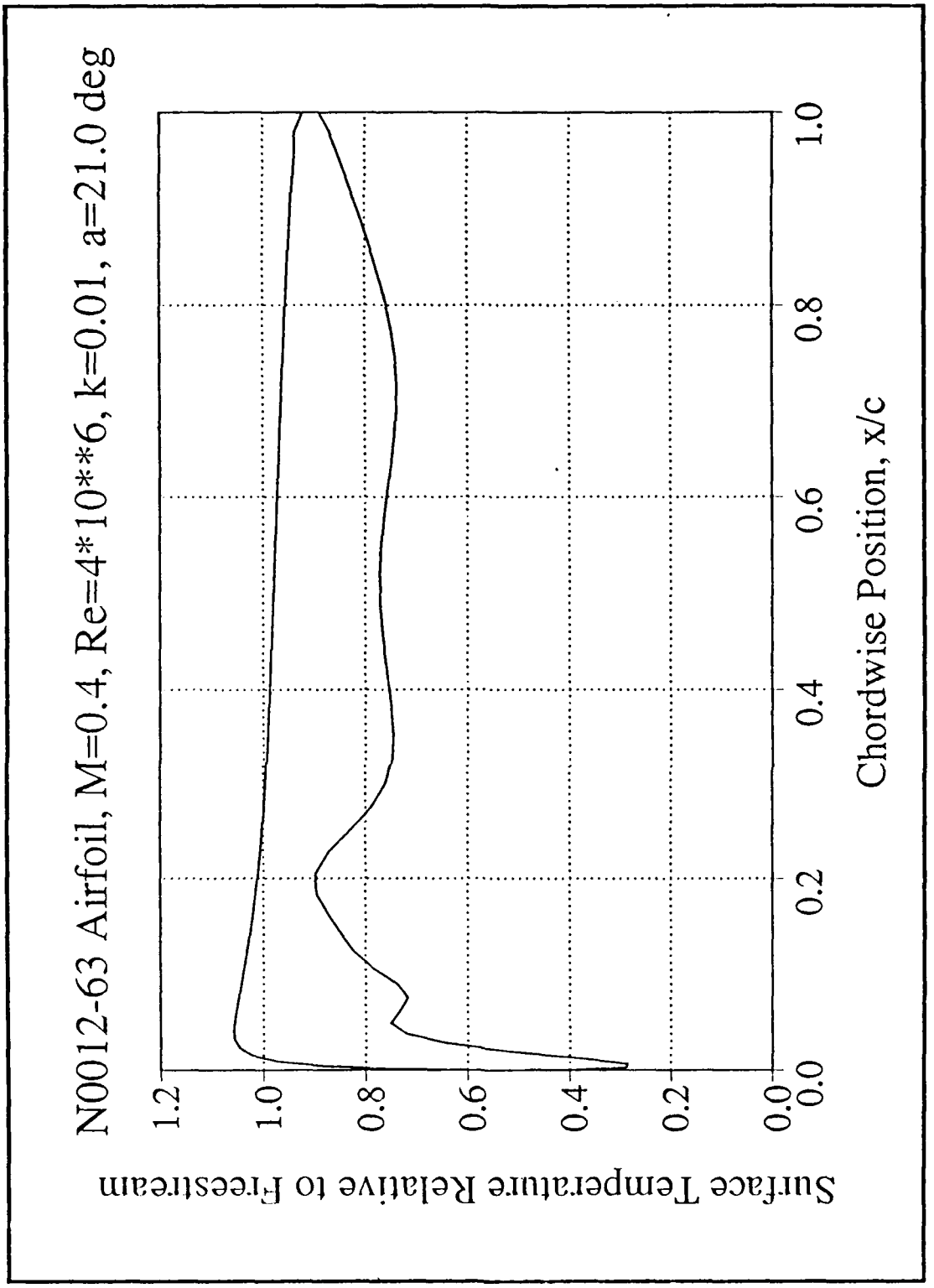

Figure 47. Surface Temperature, NACA 0012-63, M=0.4, k=0.01, α=21.0°

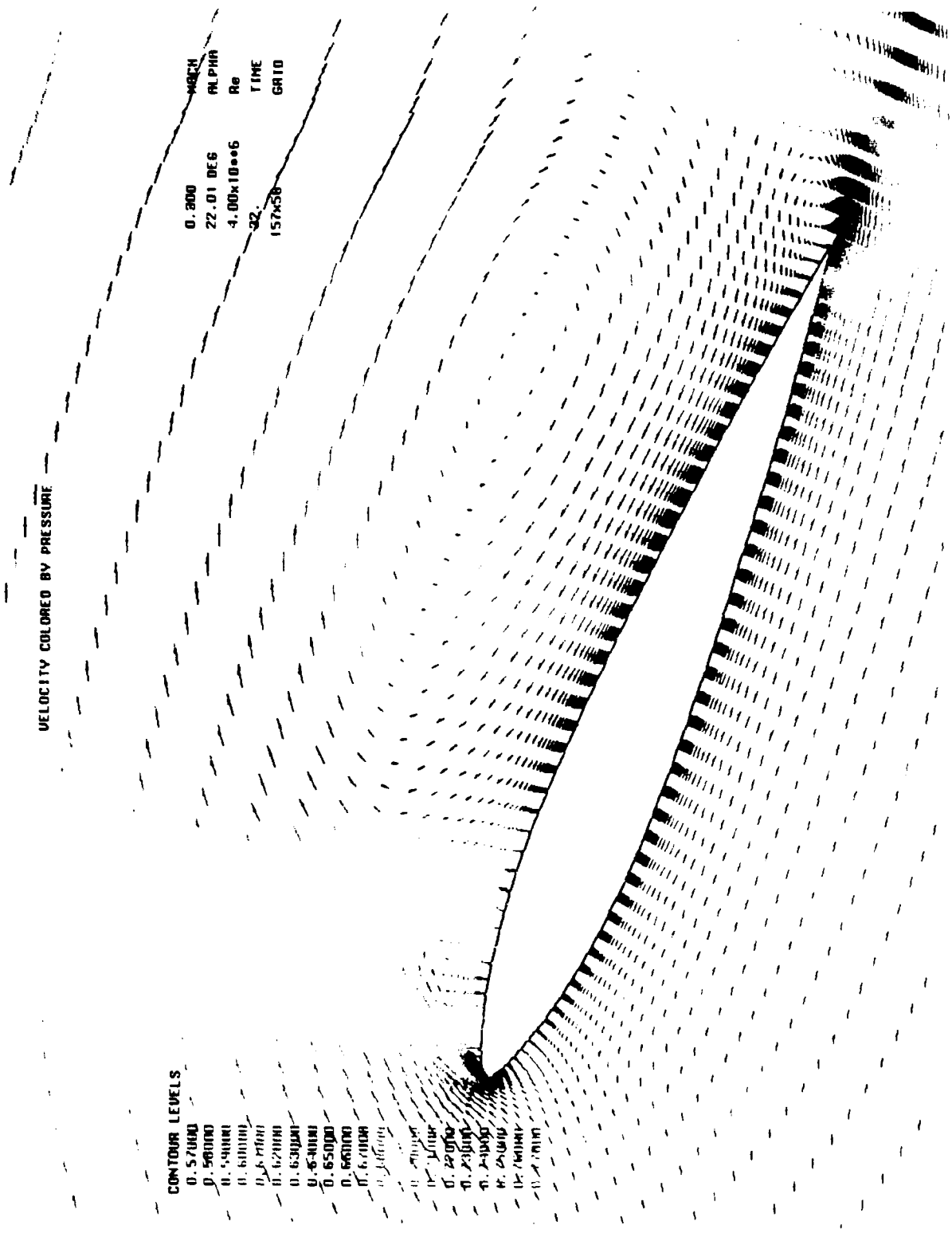

Figure 48. Velocity Field, NACA 0012-33, M=0.3, k=0.02, α=22.0°

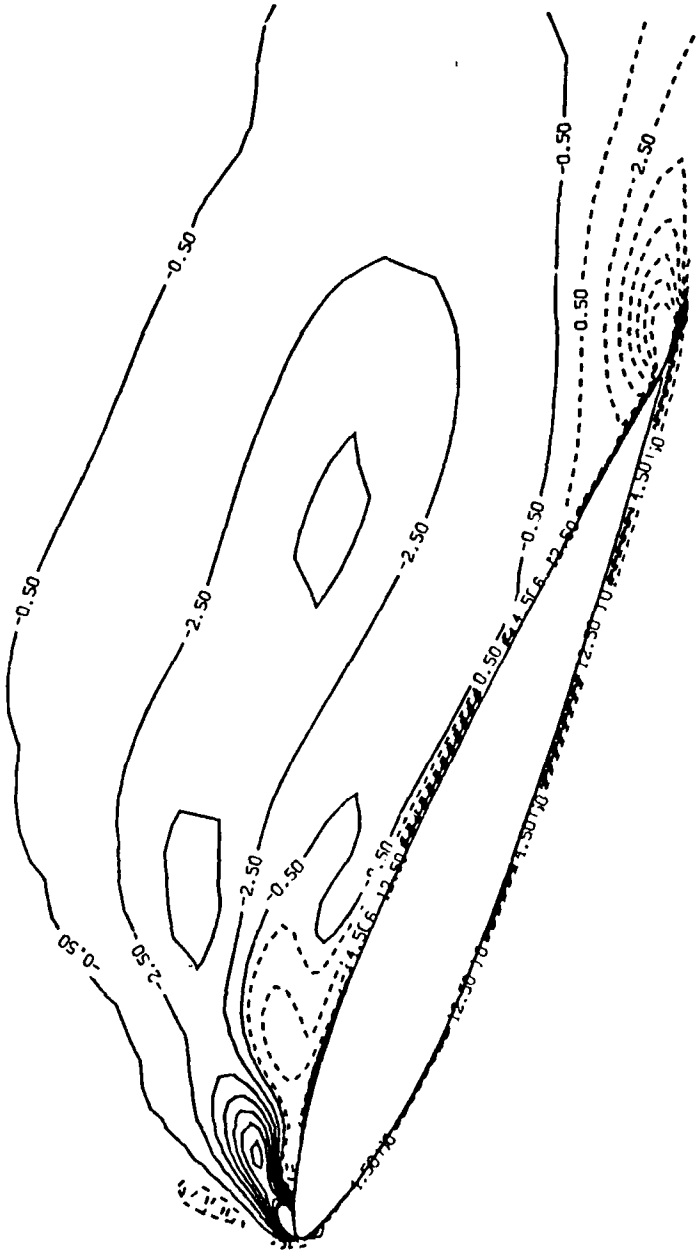

Figure 49. Vorticity Contours, NACA 0012-33, M=0.3, k=0.02, α=22.0°

2. Harmonically Oscillating Airfoil

The process of vortical flow development at the leading edge during the upstroke is similar to that of a rapidly pitching airfoil examined in the previous section. Although the pitching rate is not constant in the case of the harmonically oscillating airfoil, boundary layer separation and vortex development and shedding also occur during the upstroke.

The flow reattachment process during the downstroke involves the shedding of the large primary vortex into the wake and the diminishing intensity of the trailing edge vortex. Prior to the downstroke, the combination of the large clockwise primary vortex and the counter-clockwise trailing edge vortex cause extensive reverse flow, even outside the boundary layer, over the airfoil upper surface aft of 30% chord (Figure 50). As the angle of attack decreases further, flow over the leading edge upper surface rapidly reattaches, while the primary vortex is centered above the trailing edge (Figure 51). With further reduction in angle of attack, the primary vortex moves downstream of the airfoil, and the trailing edge vortex diminishes (Figure 52). Reverse flow at this time exists only on aft portions of the airfoil. As the angle of attack approaches 10-11°, the primary vortex has been swept downstream, the trailing edge vortex has diminished, and smooth, attached flow is established over the upper surface (Figure 53).

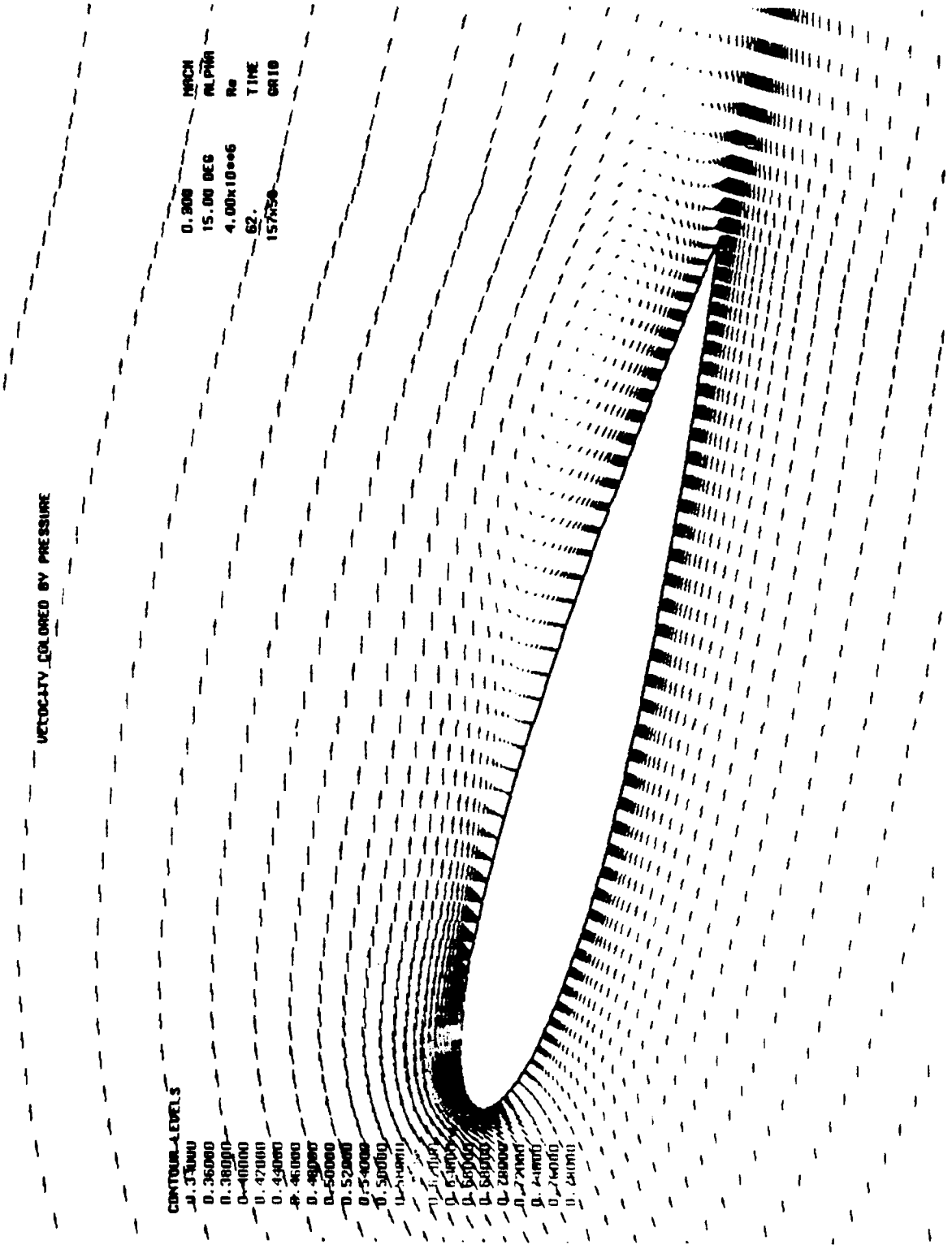

Figure 50. Velocity Field, NACA 0012, M=0.3, k=0.1, α=15.0° downstroke

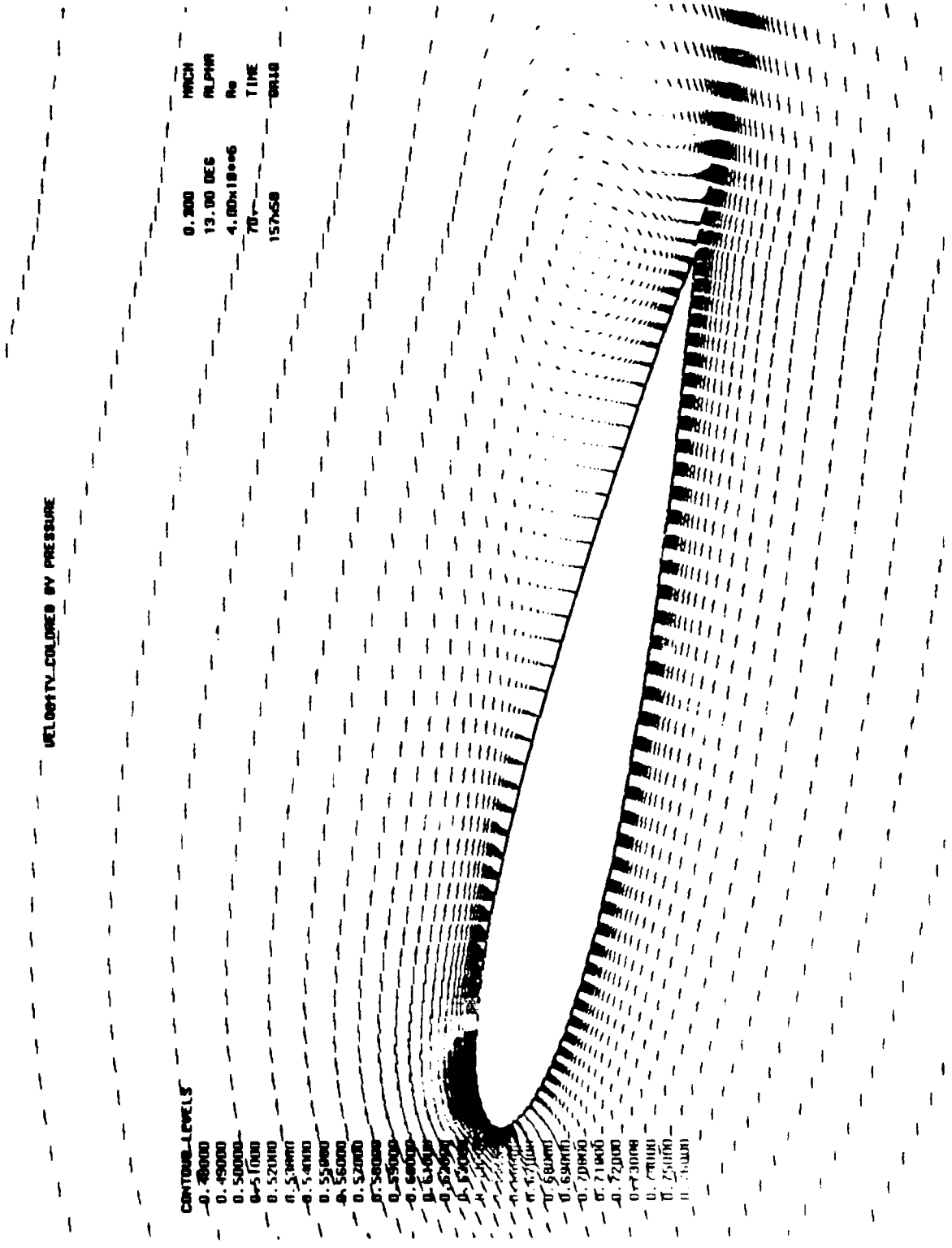

Figure 51. Velocity Field, NACA 0012, M=0.3, k=0.1, α=13.0° downstroke

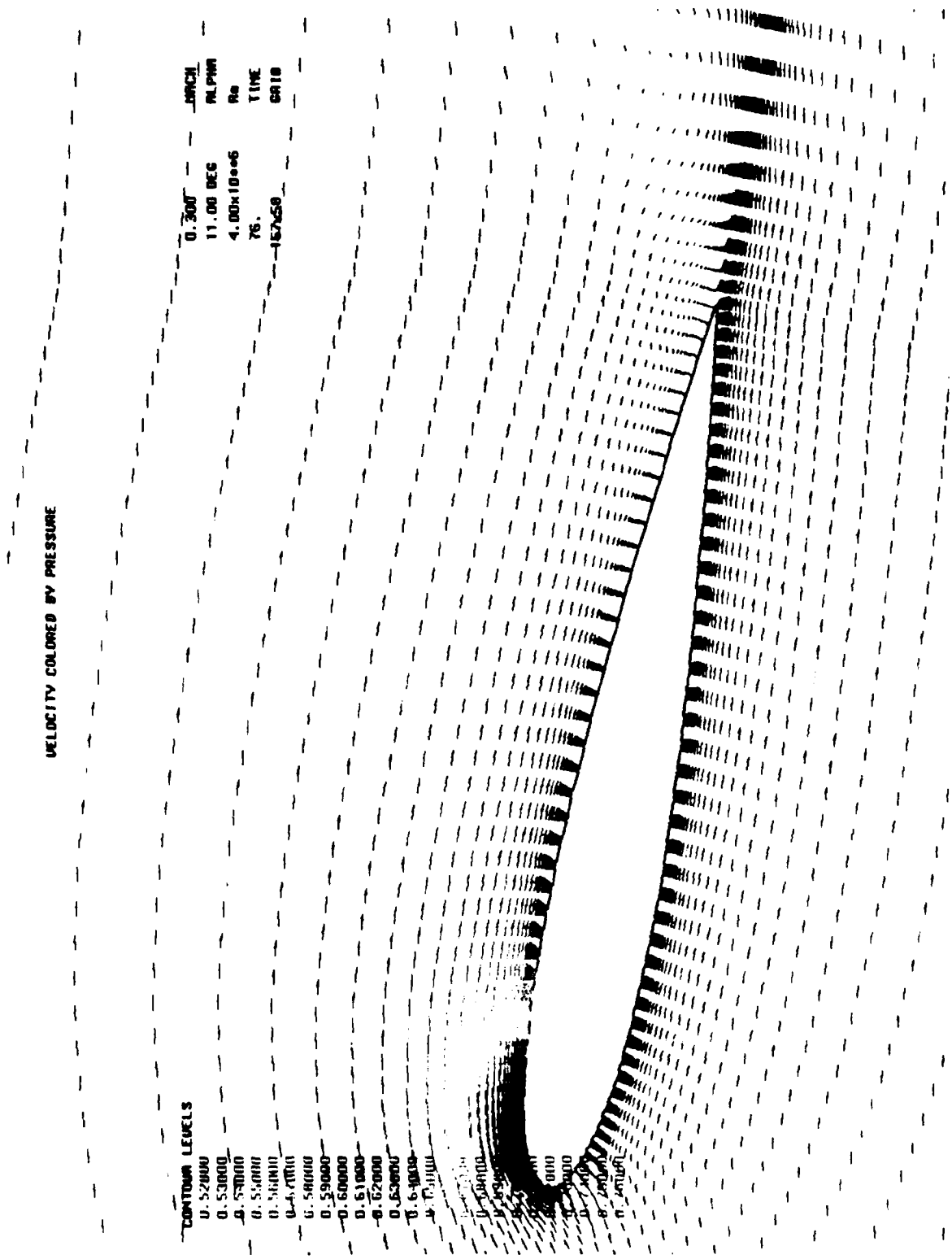

Figure 52. Velocity Field, NACA 0012, M=0.3, k=0.1, α=11.0° downstroke

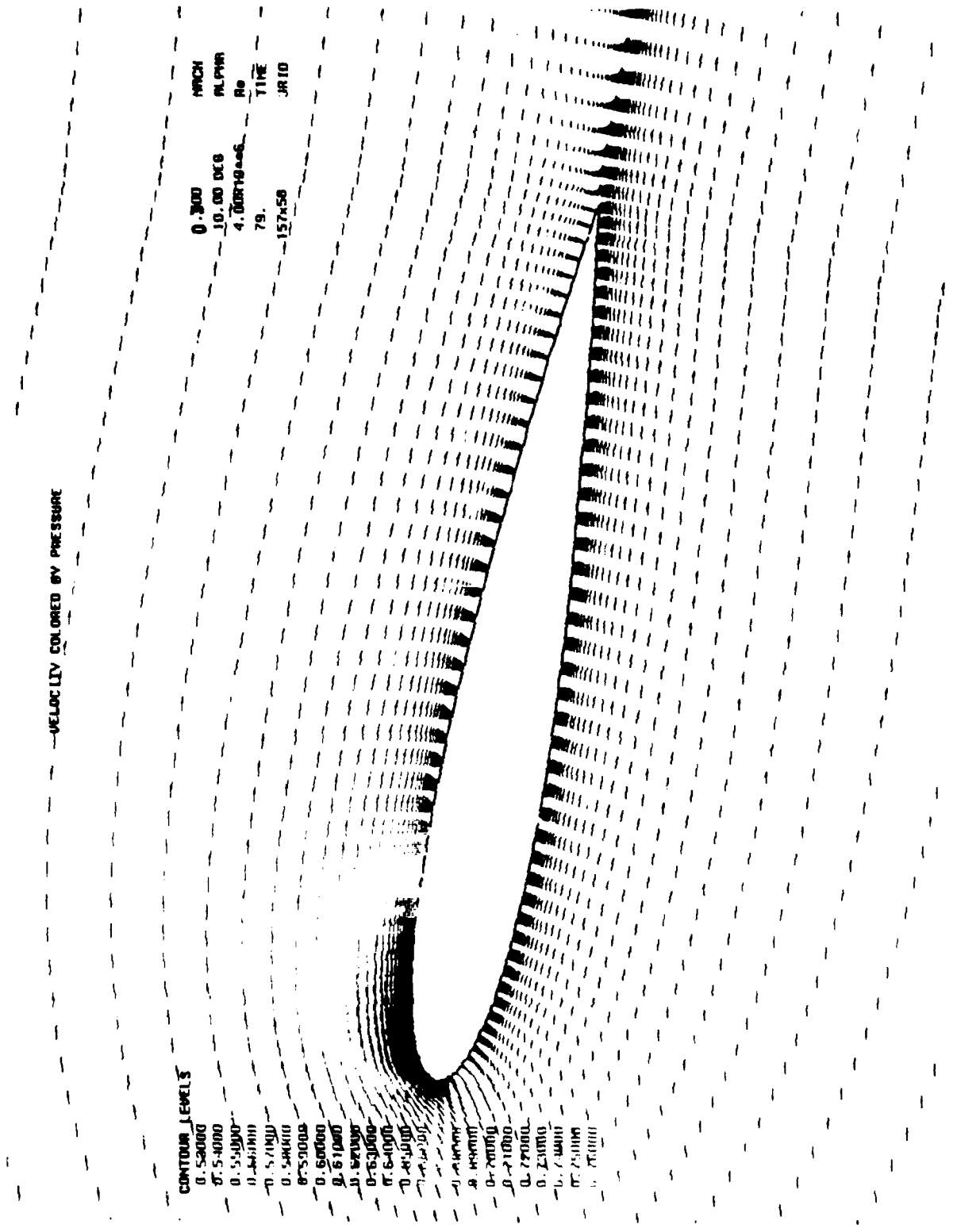

Figure 53. Velocity Field, NACA 0012, M=0.3, k=0.1, α=10.0° downstroke

D. REDUCED FREQUENCY EFFECT

At a given Mach number the three airfoils exhibited higher peak lift coefficients at respectively higher angles of attack at a reduced frequency of k=0.02 when compared to that of k=0.01. Lift curve slopes for a given airfoil at the two reduced frequencies were nearly identical to the point of initial flow separation and subsequent dynamic stall at k=0.01. Lift coefficient at k=0.02 continued increasing, showing the increase in unsteady lift with higher pitching rate. The effect of increasing the reduced frequency at a freestream Mach number of 0.3 is shown in Figures 54 and 55 for the NACA 0012 and NACA 0012-33 airfoils, respectively. At the higher reduced frequency of 0.02, formation of the primary vortex occurred at angles of attack 0.7° to 1.6° higher than at a reduced frequency of 0.01 in all cases. As a result, the entire vortical flow development was delayed to progressively higher angles of attack. Dynamic stalling angle occurred 4.0° to 4.8° higher at the higher reduced frequency and the resulting peak lift coefficients were 15-20% higher compared to the k=0.01 cases (Table 3).

Experimental work by Chandrasekhara and Carr [Ref. 3] using the NACA 0012 airfoil demonstrated the same trends, wherein a higher reduced frequency resulted in retaining the vortex above the airfoil to higher angles of attack, thereby delaying dynamic stall.

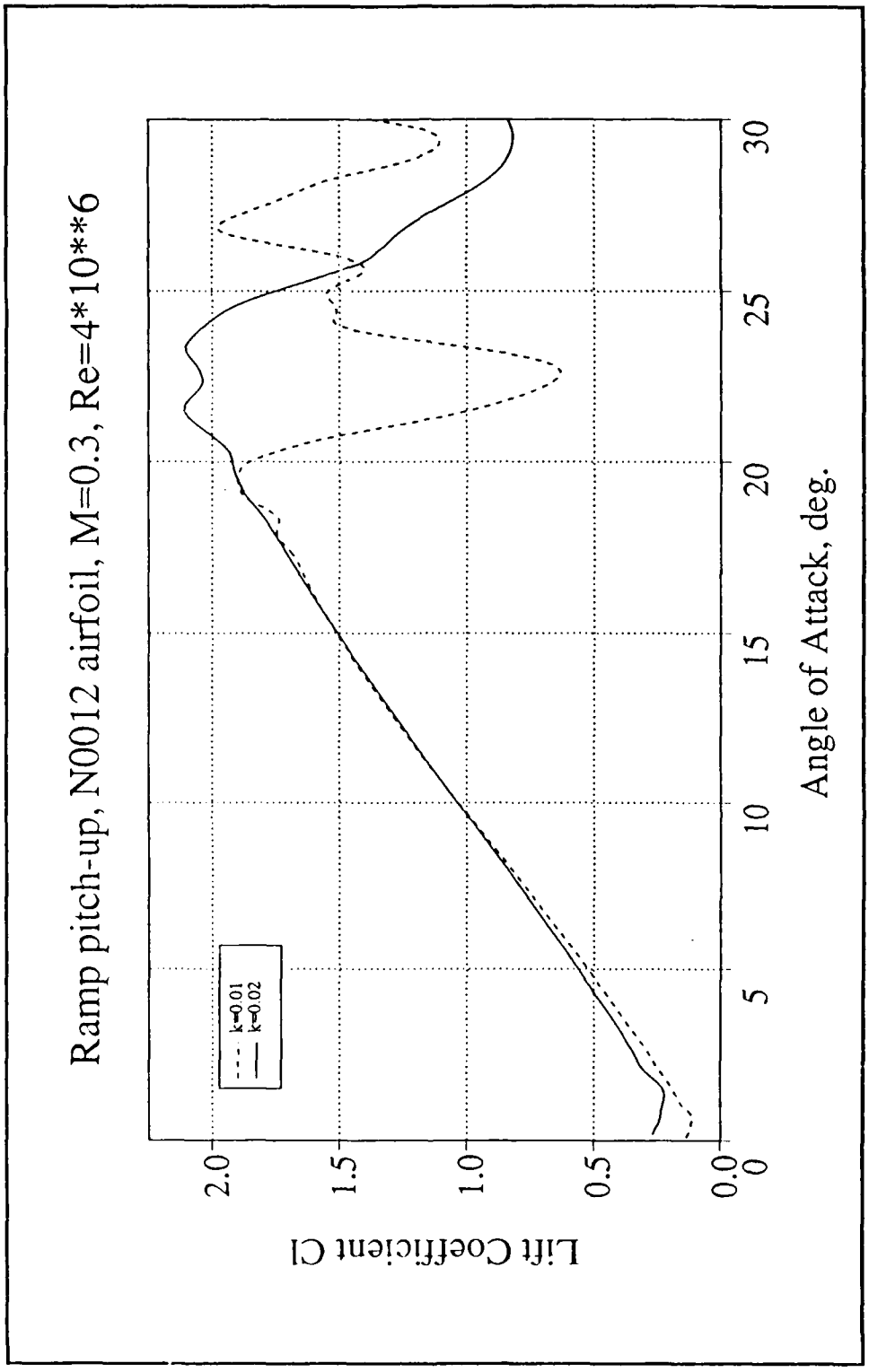

Figure 54. Reduced Frequency Effect, NACA 0012, M=0.3

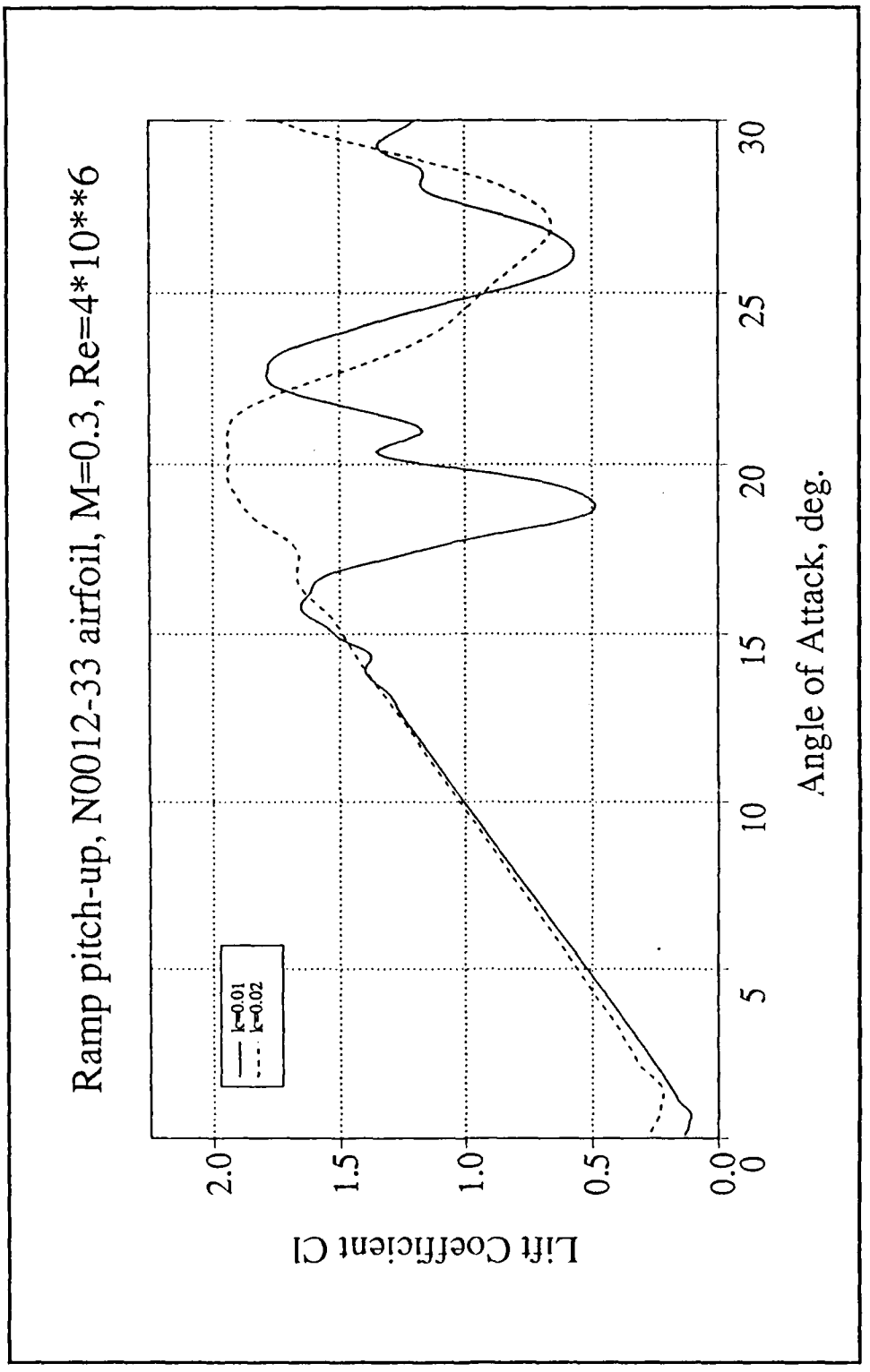

Figure 55. Reduced Frequency Effect, NACA 0012-33, M=0.3

The effect of the increased pitching rate in delaying the onset of dynamic stall and increasing the unsteady lift may prove to be of practical operational benefit as knowledge of the details of the flow development increases.

E. EFFECT OF FREESTREAM MACH NUMBER

The effect of increasing freestream Mach number from 0.3 to 0.4 resulted in slightly displacing the lift curve upward for the higher Mach number. The slope of the lift curve was unchanged. However, for a given angle of attack, a very slight increase in lift coefficient was observed at the higher Mach number before the onset of dynamic stall. Figures 56 and 57 show the effect of increasing Mach number for the NACA 0012 and 0012-33 airfoils at a reduced frequency of k=0.02. Flow at a Mach number of 0.4 resulted in slightly higher peak lift coefficients than at M=0.3. In addition, the peak lift coefficient at M=0.4 occurred 1-2° higher angle of attack than at M=0.3.

The initial development of reverse flow regions and vorticity was largely independent of Mach number for a given airfoil in that the primary vortex originated at approximately the same angle of attack for the two Mach numbers. Subsequent growth of the primary vortex and development of the secondary, tertiary, and trailing edge vortices are delayed to slightly higher angles of attack at M=0.4 compared to the M=0.3 case. As a result, dynamic stall occurs at a higher angle of attack.

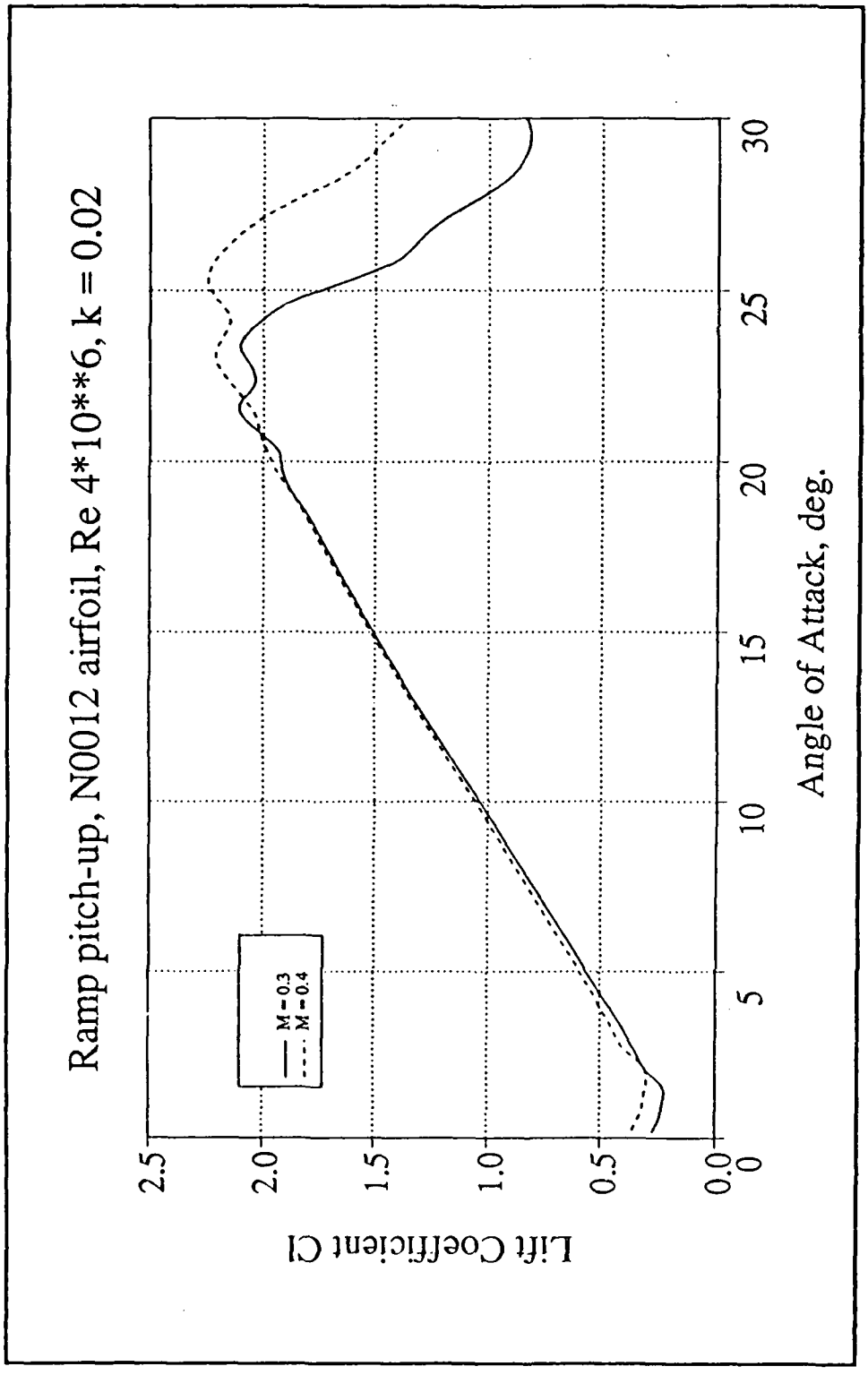

Figure 56. Freestream Speed Effect, NACA 0012, k=0.02

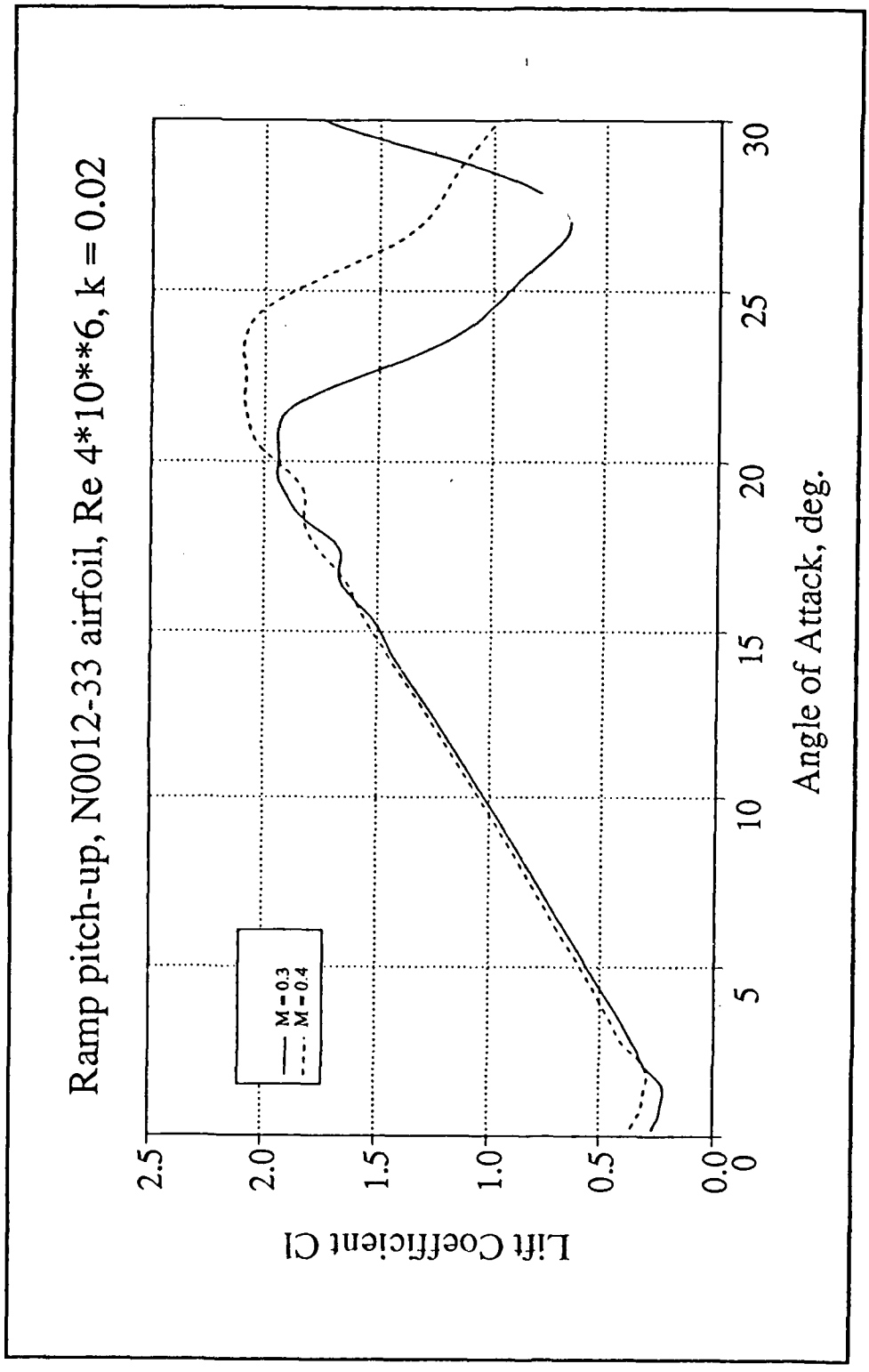

Figure 57. Freestream Speed Effect, NACA 0012-33, k=0.02

Ekaterinaris [Ref. 11] found that for the SSC-a90 airfoil with 9% thickness ratio, increasing Mach number from 0.2 to 0.4 resulted in decreasing both lift coefficient and dynamic stalling angle of attack at a reduced frequency of 0.01. The experimental work by Chandrasekhara and Carr [Ref. 3] using a harmonically oscillating NACA 0012 airfoil also showed that increasing Mach number reduces the angle of attack at which dynamic stall occurred. Further experimental and computational investigation is required at comparable pitching rates to ascertain the accuracy of the computational model used at compressible flow Mach numbers. Further investigation into the influence of airfoil thickness ratio on Mach number effects is needed.

F. PRESSURE GRADIENT AT FLOW SEPARATION

The peak pressure gradient observed at initial flow separation occurring aft of the leading edge was investigated for the three airfoils at the reduced frequencies and Mach numbers listed in Table 2. Figure 58 displays a plot of streamwise pressure gradient along the airfoil chord for the NACA 0012-63 airfoil at different freestream conditions. The figure shows that, for the same airfoil, the peak pressure gradient encountered by the flow at initial flow separation is independent of freestream speed or pitching rate. Similar results were obtained for the other airfoils. Figure 59 shows the streamwise pressure gradient for the NACA 0012-33 airfoil.

Figure 60 displays pressure gradients at initial flow separation angle of attack for the three airfoils at the freestream condition of 0.4 Mach and reduced frequency of k=0.01. The peak pressure gradient observed at the instant of flow reversal was higher for the NACA 0012-33 airfoil than for the airfoils with larger leading edge radius. Similar results were obtained at other freestream conditions. The streamwise location of the peak pressure gradient for the NACA 0012-33 airfoil was slightly upstream of that for the NACA 0012-63 airfoil as shown in Figure 61. The higher peak pressure gradient observed for the NACA 0012-33 airfoil is an indication that flow momentum near the surface is higher around the smaller leading edge radius. The higher flow momentum combined with the relatively upstream location of the peak pressure gradient and sharper flow turning angle combined to cause the magnitude of the peak pressure gradient for the NACA 0012-33 airfoil to be 33% greater than that of the NACA 0012-63 airfoil. Thus, the critical pressure gradient required for flow reversal is dependent on leading edge radius, in that a smaller leading edge radius increases the peak pressure gradient at flow reversal.

The streamwise location of flow separation is, in turn, dependent on the location of the peak pressure gradient. The actual location of flow reversal on the airfoil surface occurred 1.0-1.5% chord length downstream of the location of the peak pressure gradient as shown in Figure 61. This delay

is attributed to the time lag in the aerodynamic response to the rapid increase in pressure gradient. The more downstream location of the peak pressure gradient of the NACA 0012-63 airfoil caused a similar relative location of the position of initial flow reversal compared the NACA 0012-33 airfoil.

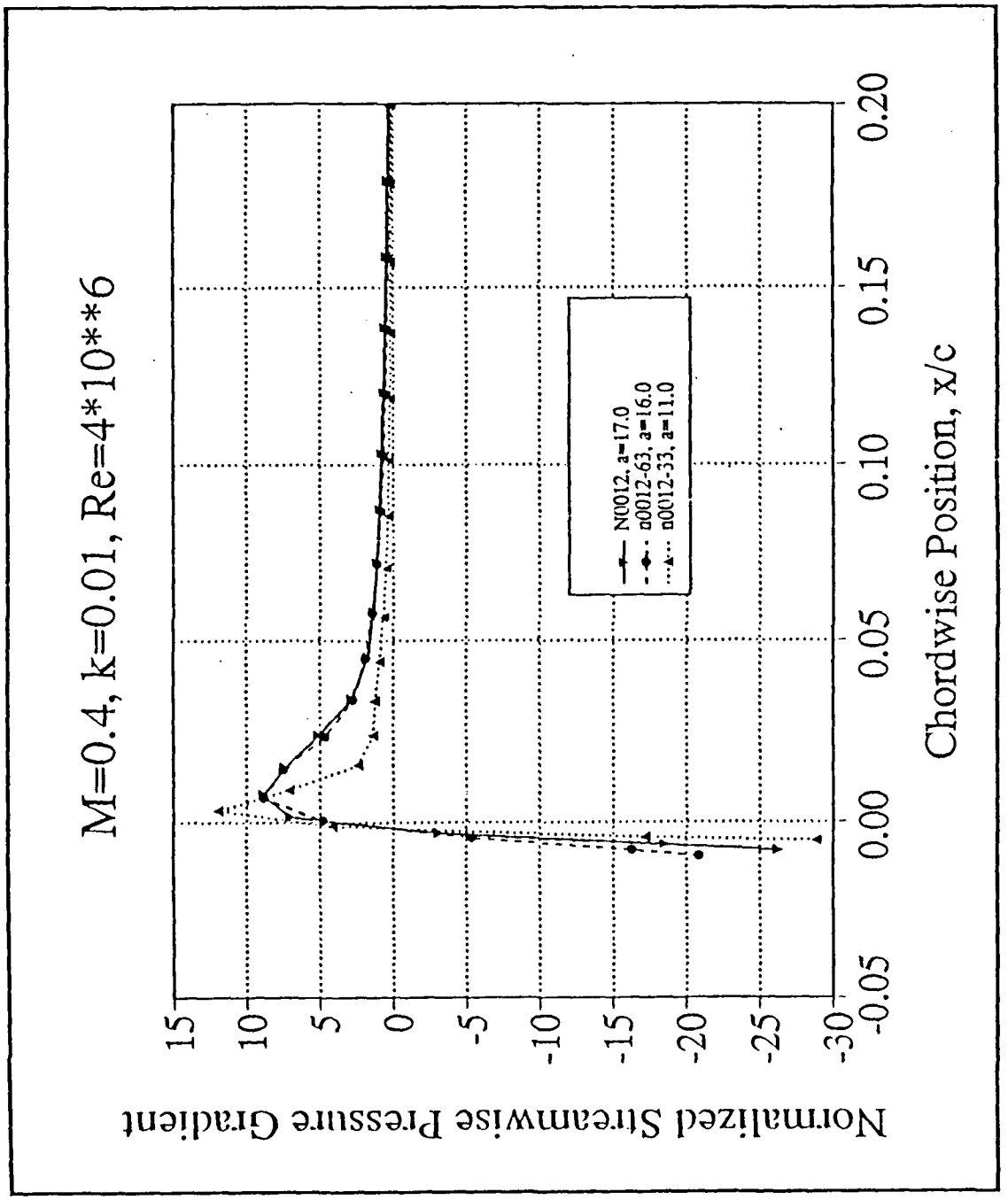

Figure 60. Pressure Gradient Comparison, M=0.4, k=0.01

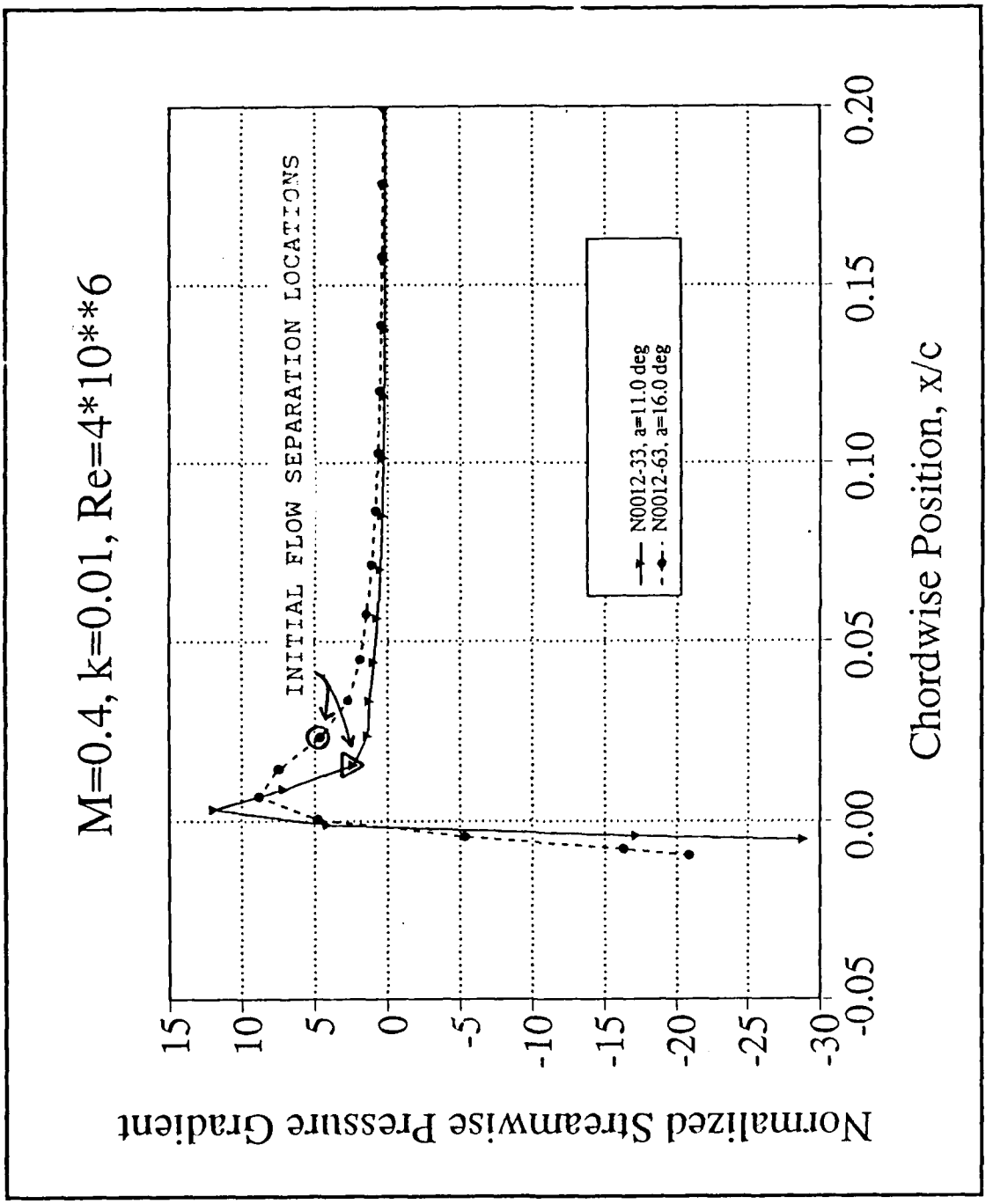

Figure 61. Flow Reversal Location, M=0.4, k=0.01

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

1. The results of the unsteady flow solutions displayed good agreement with the experimental results for the harmonically oscillating airfoil at a reduced frequency of 0.1 and Mach number of 0.3. At a higher reduced frequency of 0.2, however, the agreement with experimental results was poor during the downstroke.

2. The leading edge geometry of an airfoil was found to have a significant effect on the development of the vortical flowfield and dynamic stall characteristics of the airfoil. Of primary importance is the size of the leading edge radius. A larger leading edge radius delays development of the adverse pressure gradient necessary for boundary layer separation and eventual vortex formation. Of secondary importance is the contouring of the airfoil aft of the leading edge. Thicker contouring forward of the location of maximum thickness contributes to delaying flow separation.

3. The effect of increasing the pitching rate of the airfoil is to enhance unsteady lift by delaying vortex formation to a higher angle of attack. The flow solutions presented in this study on the effects of reduced frequency are in good agreement with trends observed in experimental work.

4. The streamwise pressure gradient required for flow separation is a function of airfoil leading edge radius and is independent of reduced frequency or freestream speed.

B. RECOMMENDATIONS

1. Conduct further experimental and computational investigations to compare Mach number effects on flow development and dynamic stall. The investigations should be conducted at comparable Mach and Reynolds numbers.

2. Investigate the effect of airfoil thickness ratio on dynamic stall.

3. The understanding of harnessing the unsteady lift generated by rapidly pitching airfoils is only part of the knowledge required for practical implementation. Further investigations must include examining pitching moment development and how to minimize the adverse pitching moments produced.

4. Conduct further computational work at higher reduced frequencies and Mach number to check the validity of using this eddy-viscosity model at those flow conditions.

LIST OF REFERENCES

1. McCroskey, W. J., *The Phenomenon of Dynamic Stall*, National Aeronautics and Space Administration Technical Memorandum 81264, March 1981.

2. McCroskey, W. J., and Pucci, S. L., *Viscous-Inviscid Interaction on Oscillating Airfoils in Subsonic Flow*, American Institute of Aeronautics and Astronautics Paper 81-0051R, January 1981.

3. Chandrasekhara, M., and Carr, L., *Flow Visualization Studies of the Mach Number Effects on the Dynamic Stall of an Oscillating Airfoil*. American Institute of Aeronautics and Astronautics Paper 89-0023, January 1989.

4. Chandrasekhara, M. and Brydges, B., "Amplitude Effects on Dynamic Stall of an Oscillating Airfoil," paper presented at the Aerospace Sciences Meeting, 28th, Reno, NV, 9-12 January 1990.

5. Mehta, U. B., *Dynamic Stall of Oscillating Airfoils*, Advisory Group for Aerospace Research and Development Paper #23 CP-227, Unsteady Aerodynamics, September 1977.

6. Wu, J. C. and others, "Dynamic Stall of Oscillating Airfoils," paper presented at 42nd Annual Forum of American Helicopter Society, Washington, D.C., June 1986.

7. Beddoes, T. S., *Prediction Methods for Unsteady Separated Flows*, Advisory Group for Aerospace Research and Development Paper #15 CP-679, 1980.

8. Jang, H. M., Ekaterinaris, J. A., Platzer, M. F., and Cebeci, T., "Essential Ingredients for the Computation of Steady and Unsteady Blade Boundary Layers," ASME Paper 90-GT-160, June 1990.

9. Sankar, N. L., and Tang, W., *Numerical Solution of Unsteady Viscous Flow for Rotor Sections*, American Institute of Aeronautics and Astronautics Paper 85-0129, January 1985.

10. Visbal, M. R., *Effect of Compressibility on Dynamic Stall of a Pitching Airfoil*, American Institute of Aeronautics and Astronautics Paper 88-0132, January 1988.

11. Ekaterinaris, J. A. *Compressible Studies on Dynamic Stall*, American Institute of Aeronautics and Astronautics Paper 89-0024, January 1989.

12. Abbott, I. H. and Von Doenhoff, A. E., *Theory of Wing Sections*, Ch. 6, Dover Publications, Inc., 1959.

13. Schlicting, H., *Boundary Layer Theory*, McGraw-Hill Book Company, 1979.

14. Vivand, H., "Conservative Forms of Gas Dynamic Equations," La Recherche Aerospatiale, No. 1 pp. 65-68, January-February 1974.

15. Beam, R. M. and Warming, R. F., "An Implicit Factored Scheme for the Compressible Navier-Stokes Equations," *American Institute of Aeronautics and Astronautics Journal*, v. 16, no. 4, pp. 393-402, April 1978.

16. Baldwin, B. S. and Lomax, H., *Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows*, American Institute of Aeronautics and Astronautics Paper 78-257, January 1978.

17. Cebeci, T., *Calculation of Compressible Turbulent Boundary Layers with Heat and Mass Transfer*, American Institute of Aeronautics and Astronautics Paper 70-741, June 1970.

18. Anderson, D. A., Tannehill, J. C., and Pletcher, R. H., *Computational Fluid Mechanics and Heat Transfer*, p. 490, Hemisphere Publishing Company, 1984.

APPENDIX A - USING THE PROGRAM

A. THE NAVIER-STOKES CODE FOR GENERATING FLOW SOLUTIONS

The main program *NSE2D* (listed in Appendix B for reference) reads input data as supplied by the input file *NSE.1d*, the grid files *fort.11* or *fort.21* as appropriate, and the flow file *fort.31*. The program forms the metrics and the Jacobian by calling the *METRIC* subroutine. After calling the *SLPS* subroutine to perform the computations, the main program outputs the rotated grid and the flow solution to the file *fort.20*, and outputs the data of lift, drag, moment and pressure coefficients, time, angle of attack, and rotational frequency to the loads file *fort.3*.

The subroutine *SLPS* is the primary subroutine, and it calls other subroutines to perform the steps in the ADI algorithm. The major subroutines that *SLPS* calls are:

Subroutine	Function
MATRIX1,MATRIX2	Form block tridiagonal matrices for the ξ and η directions
AMAT1, AMAT2	compute coefficient matrices $\partial F/\partial \xi$ and $\partial G/\partial \eta$
STRESS	supplies the viscous terms
DISSIP	adds fourth-order dissipation terms to the right-hand side of the Navier-Stokes Equations
EXPBC	applies boundary conditions

RES1	computes residuals from inviscid part of equations
EDDY	applies Baldwin-Lomax model

B. DIRECTIONS FOR EDITING INPUT FILES FOR THE PURPOSE OF GENERATING FLOW SOLUTIONS

1. Generating a Grid

Initially, a grid must be generated as described in Chapter III in order to compute the flow about the airfoil in question. For this, it is necessary to use the *AIRFGR.F* program. With the rectangular coordinates of the desired airfoil, edit the input file *AIRFGR.IN* using the "ex" or the "vi" editors. Ensure that enough points are defined in regions of high curvature such as the leading edge regions. Then ensure that no *fort.21* file exists by renaming or deleting it. At this time, run the grid program by typing

```
AIRFGR < AIRFGR.IN
```

After the program has run, ensure that the grid is satisfactory by running *PLOT3D*. It may be necessary to define more coordinate points to ensure leading edge radius definition, upper or lower surface curvature, or trailing edge closure. Accuracy of trailing edge closure will determine wake thickness. The AIRFGR.F program is listed in Appendix C for reference.

2. Rotating a Grid

The grid generated in part 1 is for a zero degree angle of attack. Often it is desired to compute a steady flow or to

start an unsteady flow at an AOA other than zero. In order to do this the *ROTGR.F* program must be used. It is necessary to edit the *ROTGR.F* program and set the desired rotated grid angle by changing the $\alpha 1$ value in line 10. After the program has been edited, then:

1. Compile *ROTGR.F* by typing "cf77 rotgr.f".
2. Compiling *ROTGR.F* results in an output file named *a.out*. Move *a.out* to *rotgr* by typing "mv a.out rotgr".
3. Execute the program by typing "rotgr".
4. The output file produced by rotgr is *fort.12*. It contains the desired grid rotation. Copy *fort.12* to *fort.11* or *fort.21* as necessary for future flow solutions.

The commands within the quotation marks should be typed, not the quotation marks themselves. Upper and lower case are not important.

3. Steady-State Solutions

Ensure that the grid is rotated to the desired angle of attack of the flow solution (part 2). The rotated grid should be copied to the file *fort.21*. The *fort.21* file must have the grid rotated to the desired angle of attack for the steady-state solution!

Modify the *NSE.IN* file as follows:

line 2:
- set desired computational time interval (DT)
- set ALFA, ALFAI, ALFA1 to zero
- set REDFRE to desired reduced frequency
- set AMINF to desired Mach number

line 8:
- set number of time steps (ISTP); usually 3000 are necessary for convergence at AOA≠0. In order to run interac-

tively and not exceed time limit, set ISTP=1000 and run three times.

line 10:
- ensure that XREF=0.25
- set TSHIFT=0.0
- set REYREF to desired Reynolds number

line 14:
- For initial run, set RESTART OSCIL RAMP as FALSE TRUE TRUE. To restart (2nd and 3rd runs), set TRUE TRUE FALSE

line 22:
- Set time to 0.0 for 1st run. Set to preceding run time for restart.

To run interactively and observe output as it develops, type:

```
NSE < NSE.IN
```

To run in the background, type:

```
NSE < NSE.IN > NSE.OUT
```

When the run is finished the last time step solution will consist of grid and flow solutions in file *fort.20*. It will be necessary to separate the grid and flow solution to restart the second run. The command "SPLIT20" will move the grid solution to *fort.11* and the flow solution to *fort.31*, and will display the final run non-dimensional time. When restarting the second run, modify NSE.IN line 14 to RESTART=TRUE. A restarted solution uses grid file *fort.11* and flow file *fort.31*. Set time in NSE.IN to time specified for the previous run when SPLIT20 was executed. Execute the next run by the desired method as before.

After convergence, save the grid and flow files (SPLIT20 to *fort.11* and *fort.31*).

4. Unsteady Solutions

Both oscillatory and ramp (constant pitch rate) solutions can be obtained by the *NSE.F* program. One must ensure that the desired steady state initial angle of attack grid and flow solutions are obtained. The input file *NSE.IN* is modified as follows:

line 2:
- set time interval, freestream Mach number and reduced frequency, where $k=\dot{\alpha}c/2U_\infty$ for ramp solution and $k=\omega c/2U_\infty$ for oscillatory solution.
- For ramp response: set ALFAI to steady-state initial conditions; ALFA, ALFA1 are unused.
- For oscillatory response of form $\alpha=\alpha_0+\alpha_1 \sin(\omega t)$, set ALFA=$\alpha_0$, ALFA1=$\alpha_1$, ALFAI=($\alpha_0-\alpha_1$) since response will start at min α.

line 8:
- set ISTP 6000-12000 depending on length of response desired.

line 10:
- set REYREF to desired Reynolds number.
- set XREF=0.25
  set TSHIFT = -0.5 so that response starts at min α which is 1/4 cycle ($-\pi/2$ time shift as $\alpha=\alpha_0 + \alpha_1 \sin(\omega t-\pi/2)$) from mean α.

line 14:
- set RESTART to TRUE
- set OSCIL to TRUE for oscillatory response (else FALSE)
- set RAMP to TRUE for ramp response (else FALSE)

line 19:
- Program flow solution outputs can be selected for various angles of attack desired. Select desired AOA solutions by listing them multiplied by 100 (i.e. 11 degrees as 1100). The outputs are listed as files *fort.61* through *fort.72*.

line 22:
- Set time to 0.0 for initial run. For restart set time to the *fort.20* output from the last timestep of the last run.

Run the program by submitting it as a job to the queue by typing "qsub -lt 7200 subunst". The subunst file is a command file which will order the steps for the Cray when the job number comes up in the queue. With at least 1 1/2 hours in computation, go run 5 miles, eat lunch, or play with the kids. Upon returning, check on queue status by typing "mqs". If no entries, the run is finished and you can split the output files and continue, if desired.

Any of the output files *fort.61* through *fort.72* and *fort.20* can be split into grid solutions and flow solutions for further analysis. *Fort.20* file must be split to continue the response at further angles of attack, and to use the run time as the next run *NSE.IN* input in line 22. For continuous ramp or oscillatory responses, it is not necessary to reset ALFA, ALFAI, ALFA1, or RESTART; it is only necessary to reset desired AOA solutions to be recorded (line 19) and the time.

5. **Example**

As an example, to run an oscillatory solution for α=10±7sin(t), it is necessary to get a steady flow solution at α=3° (3=10-7). The grid developed for the airfoil must be rotated to 3° using the *ROTGR.F* program and input to *fort.21*. At this time a steady-state solution is obtained by modifying *NSE.IN* with the following inputs:

ALFA	0.0
ALFAI	0.0
ALFA1	0.0
REDFRE	as desired
AMINF	as desired
TIME	0.0
RESTART OSCIL RAMP	FALSE TRUE FALSE
XREF	0.25
TSHIFT	0.0

Run the program by typing " NSE < NSE.IN (>NSE.OUT) "
After 1000 timesteps, issue the command SPLIT20, so that the files *fort.11* and *fort.31* will be opened. Edit *NSE.IN* with the SPLIT20 time and set RESTART=TRUE. Run the program again, and then a third time, ensuring convergence (in lift coefficient, drag coefficient, etc.) Save the final *fort.11* and *fort.31* files.

Then, modify *NSE.IN* as follows

ALFA	10.0
ALFAI	3.0
ALFA1	7.0
ISTP	6000-12000
TIME	0.0
RESTART OSCIL RAMP	TRUE TRUE FALSE
XREF	0.25
TSHIFT	-0.5

ia1, ia2, ... ia12 as desired

Run the program by issuing the command "qsub -lt 7200 subunst". After completion, save the output files (*fort.61* through *fort.72*) and *fort.20*. Do a SPLIT20, note time and AOA, then input TIME into *NSE.IN* for further runs.

Graphs of the output files may be obtained by using the plot program *PLOT3D*. Observe output files (for example *fort.65*) and graph by typing the commands: "split" and then on

the next line "65". Outputs will be in files *fort.11* (grid) and *fort.31* (flow) which are read into the *PLOT3D* input.

A typical NSE.IN file for the unsteady solution of the example problem at k=0.02 and M=0.4 would be as follows.

```
IMAX  KMAX     DT      WW    ALFA   ALFA1   ALFAI   REDFRE   AMINF
  157    58  0.005    2.00  10.00    7.00    3.00    0.02    0.40
ISPEC (FLAG FOR CHOOSING DIFFERENT SPECTRAL RADIUS)
    3
WW2X, WW2Y, WW4X, WW4Y (EXPLICIT DISSIPATION COEF. FOR X AND Y)
      0.00       0.00      0.030      0.030
      ISTP       NPER       NOUT        RES  STRUNST
    9000.    18000.      1000.       100.        0.
    REYREF       DMIN       XREF     TSHIFT
      4.00    0.00002       0.25       -0.5
 TSTAR1
  -1.0
RESTART,MULTIGRID OPTIONS SPECIFIED IN THE NEXT CARD
TRUE TRUE FALSE
CIRCOR ( CIRCULATION CORRECTION)
 TRUE
  31   127
 ITEL ITEU
 0300 0400 0500 0600 0700 0800 0900 1000 1100 1200 1300 1400
  ia1   ia2   ia3  ia4  ia5  ia6  ia7  ia8  ia9 ia10 ia11 ia12
  TIME
0.0
```

```
LINE #  SOURCE TEXT
     1  C*************************************************************
     2  C*                                                           *
     3  C*         MAIN PROGRAM                                      *
     4  C*                                                           *
     5  C*************************************************************
     6        PROGRAM NSFMAIN
     7        PARAMETER (IX=180,kx=60)
     8        COMMON/SURF/PSUR(IX)
     9        COMMON/FIX/OMEGA,HDOT
    10        COMMON/MUTUR/CMU(IX,KX)
    11        COMMON/SKINCF/CF(IX)
    12        COMMON/GRIDI/X(IX,KX),Z(IX,KX)
    13        COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
    14        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
    15        COMMON/GRID/YACOB(IX,KX)
    16        COMMON/DAMP/WW,WWI,WW2X,WW2Y,WW4X,WW4Y
    17        COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
    18        COMMON/MTRIX/ XIX(IX,KX),XI2(IX,KX),ZETAX(IX,KX),ZETAZ(IX,KX)
    19       1 ,XIT(IX,KX),ZETAT(IX,KX)
    20        COMMON/PLOT/TITLE(10),NSTPT,RESD(3000),RES,CLE(3000),CDPE(3000)
    21        DIMENSION DRHO(IX,KX)
    22        COMMON/INITI/UINF,VINF
    23        COMMON/BCLOG/CIRCOR
    24        COMMON/LOGIC/RSTRT,PITCH,RAMP
    25        LOGICAL CIRCOR
    26        LOGICAL RSTRT,PITCH,RAMP
    27        CHARACTER ITITLE*80
    28        COMMON/TITL/ITITLE
    29        COMMON/L2NORM/ RESDL2(10000)
    30        PI = 4. * ATAN(1.)
    31  C
    32  C***  PROGRAM SOLVES TWO-DIMENSIONAL VISCOUS FLOW PAST ARBITRARY
    33  C***  GEOMETRIES USING ADI PROCEDURE.
    34  C
    35  C     TAPE5 = FILE CONTAINING INPUT DATA
    36  C     TAPE6 = OUTPUT
    37  C     TAPE8 = FILE THAT SAVES THE FLOW FIELD AT THE END OF A RUN
    38  C     IF THE CURRENT RUN IS A RESTART OF A PREVIOUS RUN, THEN
    39  C     TAPE7 IS USED TO READ THE FLOW FIELD INTO MEMORY
    40  C
    41  C***  READ INPUT DATA
    42  C
    43        READ *,ITITLE
    44        READ (5,1) TITLE
    45        READ(5,2001)
    46        READ (5,2) IMAX,KMAX,DT,WW,ALFA,ALFA1,ALFAI,REDFRE,AMINF
    47  C
    48  C   ISPEC = FLAG TO SPECIFY DIFFERENT SPECTRAL RADIUS FOR SCALING
    49  C           WITH THE DISSIPATION
    50  C
    51        READ(5,2001)
    52        READ(5,2) ISPEC
    53        READ(5,2001)
    54        READ(5,2221) WW2X,WW2Y,WW4X,WW4Y
    55  C      NSTP = NO. OF TIME STEPS TO BE DONE ON THIS RUN
    56        READ(5,2001)
    57        READ (5,2221) FNSTP,FNPER, FNOUT,RES
    58        NPER = FNPER
    59        NSTP = FNSTP
    60        NOUT = FNOUT
    61  C
    62  C     REYREF= REYNOLDS NUMBER IN MILLIONS
    63  C     DNMIN = DISTANCE OF FIRST POINT OFF THE WALL
    64  C     FOR REYNOLDS NUMBERS UPTO 3 MILLION USE 0.00005
    65  C     XREF = REFERENCE VALUE AT X-AXIS
    66        READ (5,2001)
    67        READ (5,2221) REYREF,DNMIN,XREF,TSHIFT
    68        TSHIFT = TSHIFT * PI
    69        REREAL = REYREF * 1000000.
    70        REYREF = REYREF * 1.E+06
    71  C
    72  C    TSTART = TIME THAT THE CALCULATIONS HAVE BEEN ADVANCED
    73  C    UPTO THE PREVIOUS RUN. IF TSTART IS NEGATIVE THIS VALUE IS
    74  C    OBTAINED FROM THE TAPE 3.
    75        READ (5,2001)
    76        READ (5,2221) TSTART
    77   2221 FORMAT(4F10.0)
    78        READ (5,2001)
    79   2001 FORMAT(1X)
    80        READ (5,2000) RSTRT,PITCH,RAMP
    81        READ(5,2001)
    82        READ(5,2000) CIRCOR
    83   2000 FORMAT(3L5)
    84        read(5,*) ia1 ,ia2 ,ia3 ,ia4 , ia5, ia6, ia7, ia8, ia9, ia10
    85  c      @          ,ia11,ia12,ia13,ia14,ia15
    86        ialfad = 100.*alfad
    87        IF( PITCH )          DT = PI / (NPER*REDFRE*AMINF)
    88  C      NEGATIVE REYREF MEANS INVISCID FLOW
    89  C
    90  C***  PRINT OUT THE INPUT DATA
    91  C
    92        WWI = 3. * WW
    93        WRITE (6,4) TITLE
    94        WRITE (6,3) IMAX,KMAX,DT,WW,WWI,ALFA,ALFA1,ALFAI,REDFRE,AMINF,XREF
    95        WRITE (6,66) NPER, NSTP, NOUT
    96     66 FORMAT(/,2X,5HNPER=,I8,5X,5HNSTP=,I8,5X,5HNOUT=,I8,/)
    97        IF(REYREF.GT.0.) WRITE (6,3700) REYREF
    98        WRITE(6,67) WW2X, WW2Y, WW4X, WW4Y
    99     67 FORMAT(/,2X,5HWW2X=,F8.4,5X,5HWW2Y=,F8.4,5X,5HWW4X=,F8.4,5X,
   100       1           5HWW4Y=,F8.4,/)
   101        GAMMA=1.4
   102        ITEL = 31
   103        ITEU = 133
   104        ILE  = ( ITEU - ITEL ) / 2 + ITEL
   105  C
   106  ccc       CALL AIRFOL(ITEL,ITEU,ILE)
   107  ccc       IF(REYREF.GT.0.)          CALL CLUSTR(DNMIN)
   108  c
   109        READ(21)  IMAX,KMAX
   110        read(21)  (( X(I,K),I=1,IMAX),K=1,KMAX ),
   111       @          (( Z(I,K),I=1,IMAX),K=1,KMAX )
   112  C
   113  C***C STARTING CONDITIONS.
   114  C***  DENSITY NORMALISED WITH RESPECT TO ROINF
   115  C***  VELOCITIES NORMALISED WITH RESPECT TO AINF
   116  C***  TOTAL ENERGY NORMALISED WITH RESPECT TO (ROINF*AINF*AINF)
   117  C
   118        TOTEN=AMINF*AMINF*0.5+1./(GAMMA*(GAMMA-1.))
   119        ALFA    = ALFA * PI / 180.
   120        ALFAI   = ALFAI * PI / 180
```

```
LINE #  SOURCE TEXT
   121        ALMEAN  = ALFA
   122        ALFA1   = ALFA1 * PI / 180.
   123        ALFAMAX = ALFA1 + ALMEAN
   124        ALFACR  = ALFAMAX
   125  C   ALFAS IS THE ANGLE OF AIRFOIL WITH RESPECT TO FREESTREAM AT STEADY-STATE
   126  C   OR INITIAL POSITION OF UNSTEADY MOTION.
   127  C   IT SHOULD BE SET ACCORDING TO THE TYPE OF MOTION
   128        ALFAS = ALMEAN - ALFA1*COS(0.)
   129        UINF = AMINF * COS(ALFA)
   130        VINF = AMINF * SIN(ALFA)
   131  C      uinf = aminf
   132  C      vinf = 0.0
   133  C      call rotgrid( imax,kmax,alfa )
   134        DO 7 I=1,IMAX
   135        DO 7 K=1,KMAX
   136        Q1(I,K)=1.
   137        Q2(I,K)=UINF
   138        Q3(I,K)=VINF
   139        Q4(I,K)=TOTEN
   140      7 CONTINUE
   141        IF(RSTRT) THEN
   142        REWIND 11
   143        READ (11) IMAX , KMAX
   144        READ (11) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  ),
   145       &          ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
   146        rewind 31
   147        READ (31) IMAX , KMAX
   148        READ (31) FSMACH, ALFAD, REREAL, TIME
   149        READ (31) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )  ,
   150       &          ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  ),
   151       &          ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  ),
   152       &          ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
   153        ENDIF
   154  C
   155        if( time .gt. 500. ) time = 0.
   156        if( alfad .eq. 0 ) time = 0.
   157        TSTART = TIME
   158        IF(TSTART.GE.0.) TIME = TSTART
   159        IF(.NOT.(RSTRT)) TIME = 0.
   160        WRITE(6,68) TIME
   161     68 FORMAT(/,40HTHE CALCULATIONS STARTED AT TIME   T =  ,F12.4,/)
   162        ASTART = ALFA + ALFA1 * SIN( 2*REDFRE*AMINF * TIME + TSHIFT )
   163        ASTART = ASTART * ( 180. / PI )
   164        IDONE = TIME / DT
   165        WRITE(6,681) ASTART, IDONE
   166    681 FORMAT(2X,10HALFASTART=,F10.4,8X,15HITERATIONS DONE,5X,I6,/)
   167        CALL METRIC
   168        IMIN = ( ITEU - ITEL ) / 2 + ITEL
   169        ILOW = 2 * IMIN
   170        CBD = X(ITEL,1) - X(IMIN,1)
   171        NSTPT = NSTP + NSTPP
   172  C
   173  C---------> STARTING FROM A STEADY-SATE SOLUTION AT CERTAIN ANGLE   <Ao>
   174  C           GIVE A TIME SHIFT <TSHIFT> TO START FROM Ao AND SET INITIALY
   175  C           TIME=0., KEEP TSHIFT THE SAME THROUGH THE UNST. CALCUL.
   176  C
   177        IF ( PITCH ) THEN
   178         DTP    = PI / (NPER*REDFRE*AMINF)
   179         dt     = dtp
   180        endif
   181        alfacr = pi / 9.
   182        alfamax= almean + alfa1
   183        IF( RAMP ) READ (5,*) TIME
   184        DO 1000 ITN=1,NSTP
   185        TIME = TIME + DT
   186  C
   187  C     REDUCE THE TIME STEP TO HALF AT THE PEAK OF THE CYCLE
   188  C
   189        IF (PITCH) THEN
   190  C
   191        IF( ALFA .GT. ALFACR ) THEN
   192        DT = DTP * ( 1. - .5*(ALFA-ALFACR)/(ALFAMAX-ALFACR) )
   193        ELSE
   194        DT = DTP
   195        END IF
   196  C
   197        OMEGA = 2.*REDFRE*AMINF * COS( REDFRE* 2.*TIME*AMINF +TSHIFT)
   198       1                          *ALFA1
   199        ALOLD = ALMEAN + ALFA1 * SIN( 2. * REDFRE * AMINF *
   200       1(TIME - DT)  + TSHIFT )
   201        ALFA  = ALMEAN + ALFA1 * SIN( REDFRE * 2. * TIME * AMINF
   202       1                      + TSHIFT )
   203        ALFAD = ALFA * 45. / ATAN(1.)
   204        DALFA = ALFA - ALOLD
   205        CALL ROTGRID(IMAX,KMAX,DALFA)
   206        CALL METRIC
   207        END IF
   208        IF (RAMP) THEN
   209        OMEGA = 2. * REDFRE * AMINF
   210        ALOLD = OMEGA * ( TIME - DT )
   211        ALFA  = OMEGA * TIME
   212        ALFAD = ALFA * 45. / ATAN(1.)
   213        DALFA = ALFA - ALOLD
   214        CALL ROTGRID(IMAX,KMAX,DALFA)
   215        CALL METRIC
   216        END IF
   217        ALFAD = ALFA * 45. / ATAN(1.)
   218  C
   219        CALL SLPS(ITN,ISPEC)
   220  C
   221  C     APPLICATION OF BOUNDARY CONDITIONS
   222  C
   223        CALL EXPBC(CL)
   224  C
   225        CALL LOAD(CL,CDP,CDF,CM,ALFAS,XREF)
   226  C
   227  C     PRINT OUT PRESSURE AT THE SURFACE
   228  C
   229        IF(ITN/50*50.EQ.ITN) THEN
   230        WRITE (6,19)
   231        WRITE (6,33) ITN, TIME , DT
   232        IF(PITCH  OR. RAMP) WRITE (6,3500) ALFAD,OMEGA
   233  C     IF(ITN/50*500.EQ.ITN)CALL CPPLOT(ITEL,ITEU,AMINF,X(1,1),Z(1,1))
   234  C     IF(ITN/400*400.EQ.ITN) WRITE(6,12) (I,CF(I),I=1,IMAX)
   235        WRITE (6,3000) CL , CDP , CDF , CM
   236        END IF
   237        TIMPI = TIME / AMINF
   238        IF(ITN/50*50.EQ.ITN)WRITE(3,8000)TIME,ALFAD,OMEGA,CL,CDP,CDF,CM
   239  C     IF(ITN/1000*1000.EQ.ITN) THEN
   240  C       DO 100 I = IMIN , ITEU
```

```
LINE #  SOURCE TEXT
241  C        XOC = ( X(I,1) - X(IMIN,1) ) / CHD
242  C        WRITE(2,7000) XOC , PSUR(ILOW-I) , PSUR(I)
243  C        WRITE(9,7000) XOC , CF(ILOW-I) , CF(I)
244  C 100    CONTINUE
245  C      ENDIF
246  CWU TO CALCULATE THE AVERAGE VALUES OF CL, CD, CM
247        SCL = SCL + CL
248        SCDF = SCDF + CDF
249        SCDP = SCDP + CDP
250        SCM  = SCM  + CM
251  CWU TO NEXT CWU FOR USER SPECIFIED PLOTS
252  C     DO 984 K = 1 , KMAX
253  C     DO 984 I = 1 , IMAX
254  C 984 DRHO(I,K) = ABS( Q1(I,K) - DRHO(I,K) )
255        IRES = IFIX(RES)
256        IF(ITN/IRES*IRES.EQ.ITN) THEN
257          IPLOT = ( NSTPP + ITN ) / IRES
258          RESD(IPLOT) = 0.
259          CLH(IPLOT)  = CL
260          CDPH(IPLOT) = CDP
261          DO 985 K = 1 , KMAX
262          DO 985 I = 1 , IMAX
263  C 985     RESD(IPLOT) = AMAX1(RESD(IPLOT),DRHO(I,K))
264    985 RESD(IPLOT) = RESDL2(ITN)
265        ENDIF
266  CWU
267        IF(ITN/NOUT*NOUT .EQ. ITN ) THEN
268        IF( ITN .EQ. 1*NOUT ) IOUT = 31
269        IF( ITN .EQ. 2*NOUT ) IOUT = 32
270        IF( ITN .EQ. 3*NOUT ) IOUT = 33
271        IF( ITN .EQ. 4*NOUT ) IOUT = 34
272        IF( ITN .EQ. 5*NOUT ) IOUT = 35
273        IF( ITN .EQ. 6*NOUT ) IOUT = 36
274        IF( ITN .EQ. 7*NOUT ) IOUT = 37
275        IF( ITN .EQ. 8*NOUT ) IOUT = 38
276        IF( ITN .EQ. 9*NOUT ) IOUT = 39
277        IF( ITN .EQ. 10*NOUT ) IOUT = 40
278        IF( ITN .EQ. 11*NOUT ) IOUT = 41
279        IF( ITN .EQ. 12*NOUT ) IOUT = 42
280        IF( ITN .EQ. 13*NOUT ) IOUT = 43
281        IF( ITN .EQ. 14*NOUT ) IOUT = 44
282        IF( ITN .EQ. 15*NOUT ) IOUT = 45
283        IF( ITN .EQ. 16*NOUT ) IOUT = 46
284  C
285        iout = 20
286        REWIND IOUT
287        WRITE (IOUT) IMAX , KMAX
288        WRITE (IOUT) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  ),
289       &             ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
290        WRITE (IOUT) IMAX , KMAX
291        WRITE (IOUT) AMINF, ALFAD, REREAL, TIME
292        WRITE (IOUT) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  ) ,
293       &             ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  ),
294       &             ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )        ,
295       &             ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
296        REWIND IOUT
297  c
298        ENDIF
299  C
300        ialfad = 100.*alfad
301        ia1 =  100
302        ia2 =  200
303        ia3 =  300
304        ia4 =  400
305        ia5 =  500
306        ia6 =  600
307        ia7 =  700
308        ia8 =  800
309        ia9 =  900
310        ia10=  1000
311        ia11=  1050
312        ia12=  1100
313        IF( IALFAD .EQ.  ia1  .OR.
314       @    IALFAD .EQ.  ia2  .OR.
315       @    IALFAD .EQ.  ia3  .OR.
316       @    IALFAD .EQ.  ia4  .OR.
317       @    IALFAD .EQ.  ia5  .OR.
318       @    IALFAD .EQ.  ia6  .OR.
319       @    IALFAD .EQ.  ia7  .OR.
320       @    IALFAD .EQ.  ia8  .OR.
321       @    IALFAD .EQ.  ia9  .OR.
322       @    IALFAD .EQ.  ia10 .OR.
323       @    IALFAD .EQ.  ia11 .OR.
324       @    IALFAD .EQ.  ia12 ) THEN
325         IF( IALFAD .EQ.  ia1  ) IAOUT = 61
326         IF( IALFAD .EQ.  ia2  ) IAOUT = 62
327         IF( IALFAD .EQ.  ia3  ) IAOUT = 63
328         IF( IALFAD .EQ.  ia4  ) IAOUT = 64
329         IF( IALFAD .EQ.  ia5  ) IAOUT = 65
330         IF( IALFAD .EQ.  ia6  ) IAOUT = 66
331         IF( IALFAD .EQ.  ia7  ) IAOUT = 67
332         IF( IALFAD .EQ.  ia8  ) IAOUT = 68
333         IF( IALFAD .EQ.  ia9  ) IAOUT = 69
334         IF( IALFAD .EQ.  ia10 ) IAOUT = 70
335         IF( IALFAD .EQ.  ia11 ) IAOUT = 71
336         IF( IALFAD .EQ.  ia12 ) IAOUT = 72
337         REWIND IAOUT
338         WRITE (IAOUT) IMAX , KMAX
339         WRITE (IAOUT) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  ),
340        &              ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
341         WRITE (IAOUT) IMAX , KMAX
342         WRITE (IAOUT) AMINF, ALFAD, REREAL, TIME
343         WRITE (IAOUT) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  ) ,
344        &              ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  ),
345        &              ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )  ,
346        &              ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
347         REWIND IAOUT
348                      ENDIF
349    1000 CONTINUE
350  C
351        REWIND 20
352        WRITE (20) IMAX , KMAX
353        WRITE (20) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  ),
354       &           ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
355        WRITE (20) IMAX , KMAX
356        WRITE (20) AMINF, ALFAD, REREAL, TIME
357        WRITE (20) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  ),
358       &           ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  ),
359       &           ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )  ,
360       &           ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
```

```
LINE #  SOURCE TEXT
  361   C
  362         alfaf = alfa * (180/pi)
  363         write(6,667) alfaf, time
  364     667 format(17hANGLE OF ATTACK =,F15.10,5X,5HTIME=,F10.4,//)
  365   C
  366   C      PRINT OUT VELOCITY PROFILE
  367   C
  368         WRITE(6,668)
  369     668 FORMAT(/,14X,2HUX,14X,2HUZ,10X,8HDENSITY ,6X,8HPRESSURE,//)
  370         CMUL = AMINF / REYREF
  371         DO 4000 I = 70,130,2
  372         S = 0.
  373         DO 4000 K = 2 , 20
  374         S = S + SQRT( (X(I,K)-X(I,K-1))**2 + (Z(I,K)-Z(I,K-1))**2 )
  375         U = ( Q2(I,K)/Q1(I,K) )
  376         V = ( Q3(I,K)/Q1(I,K) )
  377         UTOT = U*U + V*V
  378         P = (GAMMA-1.)*( Q4(I,K) - .5*Q1(I,K)*UTOT )
  379         WRITE (6,2002) I,K, U,V , Q1(I,K) , P
  380    4000 CONTINUE
  381   C     CALL OUTRVC(REREAL)
  382         STOP
  383   C     1 FORMAT(12A6)
  384         1 FORMAT(12A8)
  385         2 FORMAT(2I5,7F10.0)
  386        11 FORMAT(7F10.0)
  387         3 FORMAT(/,5X,5HIMAX=,I10,//,5X,5HKMAX=,I10,
  388          1/,5X,5H  DT=,F20.8,/,5X,5HWW  =,F20.8,/,5X,5HWWI =,F20.8,
  389          2/,5X,5HALFA=,F20.8,/,5X,6HALFAI=,F20.8,//,5X,6HALFAI=,F20.8,/,
  390          +5X,7HREDFRE=,F18.8,/,5X,6HAMINF=,F20.8,//,5X,5HXREF=,F21.8)
  391         4  FORMAT(1H1,5X,10A6)
  392        12  FORMAT(8(I4,F10.4))
  393        19  FORMAT(/)
  394   C    22  FORMAT(2F10.6,I5)
  395        33  FORMAT(5X,5HISTP=,I5,5X,5HTIME=,F9.5,5X,3HDT=,F9.5)
  396      2002  FORMAT(2I5,5E14.6)
  397      3000  FORMAT(5X,3HCL=,F10.4,5X,4HCDP=,F10.4,5X,4HCDF=,F10.4,5X,3HCM=,
  398          +F10.4)
  399      3500  FORMAT(5X,5HALFA=,F10.4,5X,6HOMEGA=,F10.4,5X,2HH=,F10.4,5X,5HHDOT=
  400          1,F10.4)
  401      3700  FORMAT(5X,7HREYREF=,F20.4)
  402   C5000  FORMAT(2F10.6)
  403      6000  FORMAT(8F14.8)
  404      7000  FORMAT(3F16.8)
  405      8000  FORMAT(9I13.6)
  406      8001  FORMAT(3X,8HCL(AVG)=,F12.7,4X,9HCDF(AVG)=,F12.7,4X,9HCDP(AVG)=,
  407          +F12.7,4X,8HCM(AVG)=,F12.7)
  408         END
```

```
LINE #
  409       SUBROUTINE AMAT1(K,IMX1,XIX,XIZ,XIT)
  410 C*************************************************************
  411 C*                                                           *
  412 C*         SUBROUTINE AMAT1                                  *
  413 C*                                                           *
  414 C*************************************************************
  415       PARAMETER (IX=180,kx=60)
  416       COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
  417       COMMON/PERTR/DQ1(IX,KX),DQ2(IX,KX),DQ3(IX,KX),DQ4(IX,KX)
  418       COMMON/AM A(4,4,IX)
  419       COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
  420       DIMENSION XIT(IX,KX),XIX(IX,KX),XIZ(IX,KX)
  421       REAL K0,K1,K2
  422 C
  423 C***  AMAT1 COMPUTES THE COEFFICIENT MATRIX DE/DQ DURING XI SWEEP
  424 C
  425       GM1       = GAMMA - 1.
  426       DO 1000 I = 2 , IMX1
  427       K0        = XIT(I,K)
  428       K1        = XIX(I,K)
  429       K2        = XIZ(I,K)
  430       U         = Q2(I,K) / Q1(I,K)
  431       W         = Q3(I,K) / Q1(I,K)
  432       EBYR      = Q4(I,K) / Q1(I,K)
  433       PHI2      = 0.5 * GM1 * (U * U + W * W)
  434       THETA     = K1 * U + K2 * W
  435       A(1,1,I)  = K0
  436       A(1,2,I)  = K1
  437       A(1,3,I)  = K2
  438       A(1,4,I)  = 0
  439       A(2,1,I)  = K1 * PHI2 - U * THETA
  440       A(2,2,I)  = K0 + THETA - K1 * (GM1 - 1.) * U
  441       A(2,3,I)  = K2 * U - GM1 * K1 * W
  442       A(2,4,I)  = K1 * GM1
  443       A(3,1,I)  = K2 * PHI2 - W * THETA
  444       A(3,2,I)  = K1 * W - K2 * GM1 * U
  445       A(3,3,I)  = K0 + THETA - K2 * (GM1 - 1.) * W
  446       A(3,4,I)  = K2 * GM1
  447       A(4,1,I)  = THETA * (2. * PHI2 - GAMMA * EBYR)
  448       A(4,2,I)  = K1 * (GAMMA * EBYR - PHI2) - GM1 * U * THETA
  449       A(4,3,I)  = K2 * (GAMMA * EBYR - PHI2) - GM1 * W * THETA
  450       A(4,4,I)  = K0 + GAMMA * THETA
  451  1000 CONTINUE
  452       RETURN
  453       END
```

```
LINE #  SOURCE TEXT
  454       SUBROUTINE AMAT2(I,KMX1,ZETAX,ZETAZ,ZETAT)
  455   C*************************************************************
  456   C*                                                           *
  457   C*          SUBROUTINE AMAT2                                 *
  458   C*                                                           *
  459   C*************************************************************
  460       PARAMETER (IX=180,KX=60)
  461       COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
  462       COMMON/PERTR/DQ1(IX,KX),DQ2(IX,KX),DQ3(IX,KX),DQ4(IX,KX)
  463       COMMON/AM/A(4,4,IX)
  464       COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
  465       DIMENSION ZETAX(IX,KX),ZETA2(IX,KX),ZETAT(IX,KX)
  466       REAL K0,K1,K2
  467   C
  468   C***  AMAT2 COMPUTES THE COEFFICIENT MATRIX DF/DQ DURING ETA SWEEP
  469   C
  470       GM1        = GAMMA - 1.
  471       DO 1000 K = 2 , KMX1
  472       K0         = ZETAT(I,K)
  473       K1         = ZETAX(I,K)
  474       K2         = ZETAZ(I,K)
  475       U          = Q2(I,K) / Q1(I,K)
  476       W          = Q3(I,K) / Q1(I,K)
  477       EBYR       = Q4(I,K) / Q1(I,K)
  478       PHI2       = 0.5 * GM1 * (U * U + W * W)
  479       THETA      = K1 * U + K2 * W
  480       A(1,1,K)   = K0
  481       A(1,2,K)   = K1
  482       A(1,3,K)   = K2
  483       A(1,4,K)   = 0
  484       A(2,1,K)   = K1 * PHI2 -U * THETA
  485       A(2,2,K)   = K0 + THETA - K1 * (GM1-1.) * U
  486       A(2,3,K)   = K2 * U - GM1 * K1 * W
  487       A(2,4,K)   = K1 * GM1
  488       A(3,1,K)   = K2 * PHI2 - W * THETA
  489       A(3,2,K)   = K1 * W - K2 * GM1 * U
  490       A(3,3,K)   = K0 + THETA -K2 * (GM1-1.) * W
  491       A(3,4,K)   = K2 * GM1
  492       A(4,1,K)   = THETA * (2  * PHI2   GAMMA * EBYR)
  493       A(4,2,K)   = K1 * (GAMMA * EBYR - PHI2) - GM1 * U * THETA
  494       A(4,3,K)   = K2 * (GAMMA * EBYR - PHI2) - GM1 * W * THETA
  495       A(4,4,K)   = K0 + GAMMA * THETA
  496  1000 CONTINUE
  497       RETURN
  498       END
```

```
LINE #  SOURCE TEXT
  499       SUBROUTINE SLPS(ITN,ISPEC)
  500 C*****************************************************************
  501 C*                                                               *
  502 C*         SUBROUTINE SLPS                                       *
  503 C*                                                               *
  504 C*****************************************************************
  505       PARAMETER (IX=180,KX=60)
  506       COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
  507       COMMON/FIX/OMEGA,HDOT
  508       COMMON/PERTR/DQ1(IX,KX),DQ2(IX,KX),DQ3(IX,KX),DQ4(IX,KX)
  509       COMMON/AM_3/A(4,4,IX)
  510       COMMON/TRID1/DD(4,4,IX,KX)
  511       COMMON/TRID2/MM(4,4,IX,KX)
  512       COMMON/TRID3/EE(4,4,IX,KX)
  513       COMMON/TRID4/GG(4,IX,KX)
  514       COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
  515       COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
  516       COMMON/GRID/YACOB(IX,KX)
  517       COMMON/DAMP/WW,WW1,WW2X,WW2Y,WW4X,WW4Y
  518       COMMON/MTRIX/ XIX(IX,KX),XIZ(IX,KX),ZETAX(IX,KX),ZETAZ(IX,KX)
  519      1 ,XIT(IX,KX),ZETAT(IX,KX)
  520       COMMON/RADIUS/ SPECX(IX,KX),SPECY(IX,KX)
  521       COMMON/L2NORM/ RESDL2(10000)
  522       REAL MM,DELTAT(IX,KX)
  523 C
  524 C***  THE SUBROUTINE SLPS DOES THE BULK OF THE WORK  FOR THE ADI ALGORITHM.
  525 C***  IT CALLS FLUX AND COMPUTES RIGHT HAND SIDE DURING THE TWO SWEEPS,
  526 C***  ASSEMBLES THE COEFFICIENT MARICES, ADDS IMPLICIT AND EXPLICIT
  527 C***  DISSIPATION AND CALLS THE TRIDIAGONAL SOLVER TO OBTAIN THE FINAL
  528 C***  SOLUTION.
  529       IM1 = IMAX - 1
  530       IM2 = IMAX - 2
  531       KM1 = KMAX - 1
  532       KM2 = KMAX - 2
  533 C
  534 C***  THE DISSIPATION TERMS ARE CONSTRUCTED AND STORED IN THE ARRAYS DQ1,
  535 C***  DQ2,DQ3 AND DQ4.
  536 C
  537       CALL SPECF(ISPEC)
  538       CALL DISSIP
  539 C
  540 C***  ON TO DQ1,DQ2,DQ3 AND DQ4.
  541 C***  THE RIGHT HAND SIDE AT KNOWN TIME LEVEL IS NOW COMPUTED AND ADDED
  542       CALL RESI
  543 C
  544         DO 8999 K = 2 , KM1
  545         DO 8999 I = 2 , IM1
  546  8999   DELTAT(I,K) = 1.
  547       IF((REDFRE.LT.0.001).AND.ITN.EQ.1) THEN
  548         DO 1 K = 2 , KM1
  549         DO 1 I = 2 , IM1
  550     1   DELTAT(I,K) = 1.0 / ( 1. + SQRT(ABS(YACOB(I,K)) ) )
  551       ENDIF
  552 C
  553 C***  IF VISCOUS FLOW IS COMPUTED THE VISCOUS TERMS ARE ADDED TO DQ1 ETC. HERE.
  554 C
  555       IF(REYREF.  0.) CALL STRESS(ITN)
  556 C***  I-SWEEP.
  557       DTH = DT * 0.5
  558       DTW = DT * WW1
  559       DO 3 K          = 2 , KM1
  560       CALL AMAT1(K,IMAX-1,XIX,XIZ,XIT)
  561       DO 4 I1         = 1 , 4
  562       DO 4 I2         = 1 , 4
  563       DO 5 I          = 2 , IM1
  564       EE(I1,I2,I-1,K) =   A(I1,I2,I+1) * DTH * DELTAT(I,K)
  565       DD(I1,I2,I-1,K) = - A(I1,I2,I-1) * DTH * DELTAT(I,K)
  566     5 CONTINUE
  567     4 CONTINUE
  568 C
  569 C***  IMPLICIT DAMPING ADDED HERE.
  570 C
  571       DO 6 I          = 2 , IM1
  572       DT1             = SPECX(I,K) * DTW * DELTAT(I,K)
  573       DTWI            = 2. * SPECX(I,K) * YACOB(I,K) * DTW*DELTAT(I,K)
  574       DD(1,1,I-1,K) = DD(1,1,I-1,K) - DT1 * YACOB(I-1,K)
  575       DD(2,2,I-1,K) = DD(2,2,I-1,K) - DT1 * YACOB(I-1,K)
  576       DD(3,3,I-1,K) = DD(3,3,I-1,K) - DT1 * YACOB(I-1,K)
  577       DD(4,4,I-1,K) = DD(4,4,I-1,K) - DT1 * YACOB(I-1,K)
  578       EE(1,1,I-1,K) = EE(1,1,I-1,K) - DT1 * YACOB(I+1,K)
  579       EE(2,2,I-1,K) = EE(2,2,I-1,K) - DT1 * YACOB(I+1,K)
  580       EE(3,3,I-1,K) = EE(3,3,I-1,K) - DT1 * YACOB(I+1,K)
  581       EE(4,4,I-1,K) = EE(4,4,I-1,K) - DT1 * YACOB(I+1,K)
  582       MM(1,1,I-1,K) = 1. + DTWI
  583       MM(2,2,I-1,K) = 1. + DTWI
  584       MM(3,3,I-1,K) = 1. + DTWI
  585       MM(4,4,I-1,K) = 1. + DTWI
  586     6 CONTINUE
  587     3 CONTINUE
  588       DO 990 K        = 2 , KM1
  589       DO 990 I        = 2 , IM1
  590       GG(1,I-1,K)     = DQ1(I,K) * DELTAT(I,K)
  591       GG(2,I-1,K)     = DQ2(I,K) * DELTAT(I,K)
  592       GG(3,I-1,K)     = DQ3(I,K) * DELTAT(I,K)
  593       GG(4,I-1,K)     = DQ4(I,K) * DELTAT(I,K)
  594   990 CONTINUE
  595 C
  596 C     PERFORM BLOCK TRIDIAGONAL MATRIX INVERSION FOR THE ENTIRE PLANE
  597 C
  598       CALL MATRX1(IMAX,KMAX)
  599       DO 991 K        = 2 , KM1
  600       DO 991 I        = 2 , IM1
  601       DQ1(I,K)        = GG(1,I-1,K)
  602       DQ2(I,K)        = GG(2,I-1,K)
  603       DQ3(I,K)        = GG(3,I-1,K)
  604       DQ4(I,K)        = GG(4,I-1,K)
  605   991 CONTINUE
  606 C
  607 C     K-SWEEP BEGINS HERE.
  608 C
  609       DO 13 I         = 2 , IM1
  610       CALL AMAT2(I,KMAX-1,ZETAX,ZETAZ,ZETAT)
  611       DO 15 I1        = 1 , 4
  612       DO 15 I2        = 1 , 4
  613       DO 15 K         = 2 , KM1
  614       EE(I1,I2,I,K-1) =  A(I1,I2,K+1)*DTH * DELTAT(I,K)
  615       DD(I1,I2,I,K-1) = -A(I1,I2,K-1)*DTH * DELTAT(I,K)
  616    15 CONTINUE
  617 C
  618 C***  SECOND ORDER DAMPING ADDED HERE.
```

```
LINE #  SOURCE TEXT
  619   C
  620         DO 16 K           = 2 , KM1
  621         DT1               = SPECY(I,K) * DTW * DELTAT(I,K)
  622         DTWI              = 2. * SPECY(I,K) * YACOB(I,K) * DTW*DELTAT(I,K)
  623         DD(1,1,I,K-1) = DD(1,1,I,K-1) - DT1 * YACOB(I,K-1)
  624         DD(2,2,I,K-1) = DD(2,2,I,K-1) - DT1 * YACOB(I,K-1)
  625         DD(3,3,I,K-1) = DD(3,3,I,K-1) - DT1 * YACOB(I,K-1)
  626         DD(4,4,I,K-1) = DD(4,4,I,K-1) - DT1 * YACOB(I,K-1)
  627         EE(1,1,I,K-1) = EE(1,1,I,K-1) - DT1 * YACOB(I,K+1)
  628         EE(2,2,I,K-1) = EE(2,2,I,K-1) - DT1 * YACOB(I,K+1)
  629         EE(3,3,I,K-1) = EE(3,3,I,K-1) - DT1 * YACOB(I,K+1)
  630         EE(4,4,I,K-1) = EE(4,4,I,K-1) - DT1 * YACOB(I,K+1)
  631         MM(1,1,I,K-1) = 1. + DTWI
  632         MM(2,2,I,K-1) = 1. + DTWI
  633         MM(3,3,I,K-1) = 1. + DTWI
  634      16 MM(4,4,I,K-1) = 1. + DTWI
  635      13 CONTINUE
  636         DO 17 K           = 2 , KM1
  637         DO 17 I           = 2 , IM1
  638         GG(1,I,K-1)       = DQ1(I,K)
  639         GG(2,I,K-1)       = DQ2(I,K)
  640         GG(3,I,K-1)       = DQ3(I,K)
  641         GG(4,I,K-1)       = DQ4(I,K)
  642      17 CONTINUE
  643   C
  644   C     PERFORM BLOCK TRIDIAGONAL MATRIX INVERSION FOR THE ENTIRE PLANE
  645   C
  646         CALL MATRX2(IMAX,KMAX)
  647         DO 18 K           = 2 , KM1
  648         DO 18 I           = 2 , IM1
  649         DQ1(I,K)          = GG(1,I,K-1)
  650         DQ2(I,K)          = GG(2,I,K-1)
  651         DQ3(I,K)          = GG(3,I,K-1)
  652         DQ4(I,K)          = GG(4,I,K-1)
  653      18 CONTINUE
  654   C
  655   C***  UPDATE FLOW VARIABLES AT INTERIOR POINTS.
  656     967 CONTINUE
  657         RMAX              = 0.
  658         RUMAX             = 0.
  659         RVMAX             = 0.
  660         EMAX              = 0.
  661         DO 995 K          = 2 , KM1
  662         DO 19  I          = 2 , IM1
  663         Q1(I,K)           = Q1(I,K) + DQ1(I,K) * YACOB(I,K)
  664         Q2(I,K)           = Q2(I,K) + DQ2(I,K) * YACOB(I,K)
  665         Q3(I,K)           = Q3(I,K) + DQ3(I,K) * YACOB(I,K)
  666         Q4(I,K)           = Q4(I,K) + DQ4(I,K) * YACOB(I,K)
  667      19 CONTINUE
  668         DO 995 I          = 2 , IM1
  669         IF (RMAX.LT.ABS(DQ1(I,K)*YACOB(I,K))) THEN
  670         IR                = I
  671         KR                = K
  672         END IF
  673         RMAX              = AMAX1(RMAX,ABS(DQ1(I,K) * YACOB(I,K)))
  674         RUMAX             = AMAX1(RUMAX,ABS(DQ2(I,K) * YACOB(I,K)))
  675         RVMAX             = AMAX1(RVMAX,ABS(DQ3(I,K) * YACOB(I,K)))
  676         EMAX              = AMAX1(EMAX,ABS(DQ4(I,K) * YACOB(I,K)))
  677     995 CONTINUE
  678   C
  679   C     COMPUTE L2 NORM OF Q1, Q2, Q3, AND Q4 RESIDUAL
  680   C
  681         RL2 = 0.
  682         RUL2 = 0.
  683         RVL2 = 0.
  684         EL2  = 0.
  685         DO 996 K = 2 , KM1
  686         DO 996 I = 2 , IM1
  687         RL2  = RL2 + DQ1(I,K)**2
  688         RUL2 = RUL2 + DQ2(I,K)**2
  689         RVL2 = RVL2 + DQ3(I,K)**2
  690         EL2  = EL2  + DQ4(I,K)**2
  691     996 CONTINUE
  692         RL2 = RL2 / ( IM2*KM2 )
  693         RUL2 = RUL2 / ( IM2*KM2 )
  694         RVL2 = RVL2 / ( IM2*KM2 )
  695         EL2  = EL2 / ( IM2*KM2 )
  696         RL2  = SQRT(RL2)
  697         RUL2 = SQRT(RUL2)
  698         RVL2 = SQRT(RVL2)
  699         EL2  = SQRT(EL2)
  700         RESDL2(ITN) = RL2
  701         IF((ITN-1)/500*500.EQ.(ITN-1)) WRITE (6,3002)
  702         IF(ITN/50*50.EQ.ITN) WRITE (6,3001) RMAX,RUMAX,RVMAX,EMAX,IR,KR
  703         IF(ITN/50*50.EQ.ITN) WRITE (6,6004) RL2 ,RUL2 ,RVL2 ,EL2
  704         RETURN
  705   3002 FORMAT(/,5X,'DRMAX',11X,'DUMAX',11X,'DVMAX',11X,'DEMAX',9X,
  706        1'IR',3X,'KR')
  707   3001 FORMAT(4(E14.8,2X),2I5)
  708   6004 FORMAT(4(E14.8,2X)     )
  709         END
```

```
      SUBROUTINE MATRX1(IMAX,KMAX)
C*****************************************************************
C*                                                               *
C*         SUBROUTINE MATRX1                                     *
C*                                                               *
C*****************************************************************
      PARAMETER (IX=180,kx=60)
      COMMON/TRID1/DD(4,4,IX,KX)
      COMMON/TRID2/MM(4,4,IX,KX)
      COMMON/TRID3/EE(4,4,IX,KX)
      COMMON/TRID4/ GG(4,IX,KX)
      COMMON/CRAY/A(4,4,IX),BB(4,4,IX,KX),C(4,5,IX)
      REAL MM
      REAL L11,L21,L31,L41,L22,L32,L42,L33,L43,L44
     2,L1I,L2I,L3I,L4I
C
C     THIS SUBROUTINE PERFORMS THE BLOCK TRIDIAGONAL MATRIX IVERSION FOR
C     AN ENTIRE  PLANE DURING THE XI- SWEEP
C
      KM1 = KMAX - 1
      DO 1 I1 = 1, 4
      DO 1 K = 2 , KM1
      AI = 1. / MM(1,1,1,K)
      GG(I1,1,K) = GG(I1,1,K) * AI
      BB(I1,1,1,K) = EE(I1,1,1,K) * AI
      BB(I1,2,1,K) = EE(I1,2,1,K) * AI
      BB(I1,3,1,K) = EE(I1,3,1,K) * AI
      BB(I1,4,1,K) = EE(I1,4,1,K) * AI
    1 CONTINUE
C
      DO 1000 I = 2 , IMAX - 2
      DO 5    I1= 1 , 4
      DO 2    K = 2 , KM1
      C(I1,1,K)   = GG(I1,I,K) - DD(I1,1,I,K) * GG(1,I-1,K)
     1                         - DD(I1,2,I,K) * GG(2,I-1,K)
     2                         - DD(I1,3,I,K) * GG(3,I-1,K)
     3                         - DD(I1,4,I,K) * GG(4,I-1,K)
    2 CONTINUE
      DO 5   I2 = 1 , 4
      DO 5   K  = 2 , KM1
      A(I1,I2,K)= MM(I1,I2,I,K) - DD(I1,1,I,K) * BB(1,I2,I-1,K)
     1                          - DD(I1,2,I,K) * BB(2,I2,I-1,K)
     2                          - DD(I1,3,I,K) * BB(3,I2,I-1,K)
     3                          - DD(I1,4,I,K) * BB(4,I2,I-1,K)
      C(I1,I2+1,K)= EE(I1,I2,I,K)
    5 CONTINUE
      DO 3 K = 2 , KM1
      L11 = A(1,1,K)
      L1I = 1. / L11
      U12 = A(1,2,K) * L1I
      U13 = A(1,3,K) * L1I
      U14 = A(1,4,K) * L1I
      L21 = A(2,1,K)
      L31 = A(3,1,K)
      L41 = A(4,1,K)
      L22 = A(2,2,K) - L21 * U12
      L2I = 1. / L22
      U23 = (A(2,3,K) - L21 * U13) * L2I
      U24 = (A(2,4,K) - L21 * U14) * L2I
      L32 = A(3,2,K) - L31 * U12
      L42 = A(4,2,K) - L41 * U12
      L33 = A(3,3,K) - L31 * U13 - L32 * U23
      L3I = 1. / L33
      U34 = (A(3,4,K) - L31 * U14 - L32 * U24) * L3I
      L43 = A(4,3,K) - L41 * U13 - L42 * U23
      L44 = A(4,4,K) - L41 * U14 - L42 * U24 - L43 * U34
      L4I = 1. / L44
      C(1,1,K) = C(1,1,K) * L1I
      C(2,1,K) = (C(2,1,K) - L21 * C(1,1,K)) * L2I
      C(3,1,K) = (C(3,1,K) - L31 * C(1,1,K)
     1                     - L32 * C(2,1,K)) * L3I
      C(4,1,K) = (C(4,1,K) - L41 * C(1,1,K) - L42 * C(2,1,K)
     1                                      - L43 * C(3,1,K)) * L4I
      C(1,2,K) = C(1,2,K) * L1I
      C(2,2,K) = (C(2,2,K) - L21 * C(1,2,K)) * L2I
      C(3,2,K) = (C(3,2,K) - L31 * C(1,2,K)
     1                     - L32 * C(2,2,K)) * L3I
      C(4,2,K) = (C(4,2,K) - L41 * C(1,2,K) - L42 * C(2,2,K)
     1                                      - L43 * C(3,2,K)) * L4I
      C(1,3,K) = C(1,3,K) * L1I
      C(2,3,K) = (C(2,3,K) - L21 * C(1,3,K)) * L2I
      C(3,3,K) = (C(3,3,K) - L31 * C(1,3,K)
     1                     - L32 * C(2,3,K)) * L3I
      C(4,3,K) = (C(4,3,K) - L41 * C(1,3,K) - L42 * C(2,3,K)
     1                                      - L43 * C(3,3,K)) * L4I
      C(1,4,K) = C(1,4,K) * L1I
      C(2,4,K) = (C(2,4,K) - L21 * C(1,4,K)) * L2I
      C(3,4,K) = (C(3,4,K) - L31 * C(1,4,K)
     1                     - L32 * C(2,4,K)) * L3I
      C(4,4,K) = (C(4,4,K) - L41 * C(1,4,K) - L42 * C(2,4,K)
     1                                      - L43 * C(3,4,K)) * L4I
      C(1,5,K) =  C(1,5,K) * L1I
      C(2,5,K) = (C(2,5,K) - L21 * C(1,5,K)) * L2I
      C(3,5,K) = (C(3,5,K) - L31 * C(1,5,K)
     1                     - L32 * C(2,5,K)) * L3I
      C(4,5,K) = (C(4,5,K) - L41 * C(1,5,K) - L42 * C(2,5,K)
     1                                      - L43 * C(3,5,K)) * L4I
      C(3,1,K) = C(3,1,K) - U34 * C(4,1,K)
      C(2,1,K) = C(2,1,K) - U24 * C(4,1,K)
     1                    - U23 * C(3,1,K)
      C(1,1,K) = C(1,1,K) - U14 * C(4,1,K)
     1                    - U13 * C(3,1,K) - U12 * C(2,1,K)
      C(3,2,K) = C(3,2,K) - U34 * C(4,2,K)
      C(2,2,K) = C(2,2,K) - U24 * C(4,2,K)
     1                    - U23 * C(3,2,K)
      C(1,2,K) = C(1,2,K) - U14 * C(4,2,K)
     1                    - U13 * C(3,2,K) - U12 * C(2,2,K)
      C(3,3,K) = C(3,3,K) - U34 * C(4,3,K)
      C(2,3,K) = C(2,3,K) - U24 * C(4,3,K)
     1                    - U23 * C(3,3,K)
      C(1,3,K) = C(1,3,K) - U14 * C(4,3,K)
     1                    - U13 * C(3,3,K) - U12 * C(2,3,K)
      C(3,4,K) = C(3,4,K) - U34 * C(4,4,K)
      C(2,4,K) = C(2,4,K) - U24 * C(4,4,K)
     1                    - U23 * C(3,4,K)
      C(1,4,K) = C(1,4,K) - U14 * C(4,4,K)
     1                    - U13 * C(3,4,K) - U12 * C(2,4,K)
      C(3,5,K) = C(3,5,K) - U34 * C(4,5,K)
      C(2,5,K) = C(2,5,K) - U24 * C(4,5,K)
     1                    - U23 * C(3,5,K)
```

LINE # | SOURCE TEXT

```
      C(1,5,K) = C(1,5,K) - U14 * C(4,5,K)
     1                    - U13 * C(3,5,K) - U12 * C(2,5,K)
    3 CONTINUE
C
      DO 6 I1       = 1 , 4
      DO 9 K        = 2 , KM1
    9 GG(I1,I,K)    = C(I1,1,K)
      DO 6 I2       = 1 , 4
      DO 6 K        = 2 , KM1
      HH(I1,I2,I,K) = C(I1,I2+1,K)
    6 CONTINUE
 1000 CONTINUE
C
C     BACKWARD SUBSTITUTION
C
      DO 7 I      = IMAX - 3, 1 , - 1
      DO 7 I1     = 1 , 4
      DO 7 K      = 2 , KM1
      GG(I1,I,K) = GG(I1,I,K) - HH(I1,1,I,K) * GG(1,I+1,K)
     1                        - HH(I1,2,I,K) * GG(2,I+1,K)
     2                        - HH(I1,3,I,K) * GG(3,I+1,K)
     3                        - HH(I1,4,I,K) * GG(4,I+1,K)
    7 CONTINUE
      RETURN
      END
```

```
LINE #  SOURCE TEXT
  855       SUBROUTINE MATRI2(IMAX,KMAX)
  856   C*****************************************************************
  857   C*                                                               *
  858   C*          SUBROUTINE MATRX2                                    *
  859   C*                                                               *
  860   C*****************************************************************
  861         PARAMETER (IX=180,kx=60)
  862         COMMON/TRID1/DD(4,4,IX,KX)
  863         COMMON/TRID2/MM(4,4,IX,KX)
  864         COMMON/TRID3/EE(4,4,IX,KX)
  865         COMMON/TRID4/ GG(4,IX,KX)
  866         COMMON/SCRAT/A(4,4,IX),BB(4,4,IX,KX),C(4,5,IX)
  867         REAL MM
  868         REAL L11,L21,L31,L41,L22,L32,L42,L33,L43,L44
  869        2,L1I,L2I,L3I,L4I
  870   C
  871   C     THIS SUBROUTINE PERFORMS THE BLOCK TRIDIAGONAL MATRIX IVERSION FOR
  872   C     AN ENTIRE J=CONSTANT PLANE DURING THE ZETA- SWEEP
  873   C
  874         KM1 = KMAX - 1
  875         IM1 = IMAX - 1
  876         DO 1 I1       = 1, 4
  877         DO 1 I        = 2 , IM1
  878         AI            = 1. / MM(1,1,I,1)
  879         GG(I1,I,1)    = GG(I1,I,1) * AI
  880         BB(I1,1,I,1) = EE(I1,1,I,1) * AI
  881         BB(I1,2,I,1) = EE(I1,2,I,1) * AI
  882         BB(I1,3,I,1) = EE(I1,3,I,1) * AI
  883         BB(I1,4,I,1) = EE(I1,4,I,1) * AI
  884       1 CONTINUE
  885   C
  886         DO 1000 K = 2 , KMAX - 2
  887         DO 5    I1= 1 , 4
  888         DO 2    I = 2 , IM1
  889         C(I1,1,I) = GG(I1,I,K) - DD(I1,1,I,K) * GG(1,I,K-1)
  890        1                       - DD(I1,2,I,K) * GG(2,I,K-1)
  891        2                       - DD(I1,3,I,K) * GG(3,I,K-1)
  892        3                       - DD(I1,4,I,K) * GG(4,I,K-1)
  893       2 CONTINUE
  894         DO 5   I2 = 1 , 4
  895         DO 5   I  = 2 , IM1
  896         A(I1,I2,I)= MM(I1,I2,I,K) - DD(I1,1,I,K) * BB(1,I2,I,K-1)
  897        1                          - DD(I1,2,I,K) * BB(2,I2,I,K-1)
  898        2                          - DD(I1,3,I,K) * BB(3,I2,I,K-1)
  899        3                          - DD(I1,4,I,K) * BB(4,I2,I,K-1)
  900         C(I1,I2+1,I)= EE(I1,I2,I,K)
  901       5 CONTINUE
  902         DO 3 I = 2 , IM1
  903         L11 = A(1,1,I)
  904         L1I = 1. / L11
  905         U12 = A(1,2,I) * L1I
  906         U13 = A(1,3,I) * L1I
  907         U14 = A(1,4,I) * L1I
  908         L21 = A(2,1,I)
  909         L31 = A(3,1,I)
  910         L41 = A(4,1,I)
  911         L22 = A(2,2,I) - L21 * U12
  912         L2I = 1. / L22
  913         U23 = (A(2,3,I) - L21 * U13) * L2I
  914         U24 = (A(2,4,I) - L21 * U14) * L2I
  915         L32 = A(3,2,I) - L31 * U12
  916         L42 = A(4,2,I) - L41 * U12
  917         L33 = A(3,3,I) - L31 * U13 - L32 * U23
  918         L3I = 1. / L33
  919         U34 = (A(3,4,I) - L31 * U14 - L32 * U24) * L3I
  920         L43 = A(4,3,I) - L41 * U13 - L42 * U23
  921         L44 = A(4,4,I) - L41 * U14 - L42 * U24 - L43 * U34
  922         L4I = 1. / L44
  923         C(1,1,I) = C(1,1,I) * L1I
  924         C(2,1,I) = (C(2,1,I) - L21 * C(1,1,I)) * L2I
  925         C(3,1,I) = (C(3,1,I) - L31 * C(1,1,I)
  926        1                     - L32 * C(2,1,I)) * L3I
  927         C(4,1,I) = (C(4,1,I) - L41 * C(1,1,I) - L42 * C(2,1,I)
  928        1                                      - L43 * C(3,1,I)) * L4I
  929         C(1,2,I) = C(1,2,I) * L1I
  930         C(2,2,I) = (C(2,2,I) - L21 * C(1,2,I)) * L2I
  931         C(3,2,I) = (C(3,2,I) - L31 * C(1,2,I)
  932        1                     - L32 * C(2,2,I)) * L3I
  933         C(4,2,I) = (C(4,2,I) - L41 * C(1,2,I) - L42 * C(2,2,I)
  934        1                                      - L43 * C(3,2,I)) * L4I
  935         C(1,3,I) = C(1,3,I) * L1I
  936         C(2,3,I) = (C(2,3,I) - L21 * C(1,3,I)) * L2I
  937         C(3,3,I) = (C(3,3,I) - L31 * C(1,3,I)
  938        1                     - L32 * C(2,3,I)) * L3I
  939         C(4,3,I) = (C(4,3,I) - L41 * C(1,3,I) - L42 * C(2,3,I)
  940        1                                      - L43 * C(3,3,I)) * L4I
  941         C(1,4,I) = C(1,4,I) * L1I
  942         C(2,4,I) = (C(2,4,I) - L21 * C(1,4,I)) * L2I
  943         C(3,4,I) = (C(3,4,I) - L31 * C(1,4,I)
  944        1                     - L32 * C(2,4,I)) * L3I
  945         C(4,4,I) = (C(4,4,I) - L41 * C(1,4,I) - L42 * C(2,4,I)
  946        1                                      - L43 * C(3,4,I)) * L4I
  947         C(1,5,I) = C(1,5,I) * L1I
  948         C(2,5,I) = (C(2,5,I) - L21 * C(1,5,I)) * L2I
  949         C(3,5,I) = (C(3,5,I) - L31 * C(1,5,I)
  950        1                     - L32 * C(2,5,I)) * L3I
  951         C(4,5,I) = (C(4,5,I) - L41 * C(1,5,I) - L42 * C(2,5,I)
  952        1                                      - L43 * C(3,5,I)) * L4I
  953         C(3,1,I) = C(3,1,I) - U34 * C(4,1,I)
  954         C(2,1,I) = C(2,1,I) - U24 * C(4,1,I)
  955        1                    - U23 * C(3,1,I)
  956         C(1,1,I) = C(1,1,I) - U14 * C(4,1,I)
  957        1                    - U13 * C(3,1,I) - U12 * C(2,1,I)
  958         C(3,2,I) = C(3,2,I) - U34 * C(4,2,I)
  959         C(2,2,I) = C(2,2,I) - U24 * C(4,2,I)
  960        1                    - U23 * C(3,2,I)
  961         C(1,2,I) = C(1,2,I) - U14 * C(4,2,I)
  962        1                    - U13 * C(3,2,I) - U12 * C(2,2,I)
  963         C(3,3,I) = C(3,3,I) - U34 * C(4,3,I)
  964         C(2,3,I) = C(2,3,I) - U24 * C(4,3,I)
  965        1                    - U23 * C(3,3,I)
  966         C(1,3,I) = C(1,3,I) - U14 * C(4,3,I)
  967        1                    - U13 * C(3,3,I) - U12 * C(2,3,I)
  968         C(3,4,I) = C(3,4,I) - U34 * C(4,4,I)
  969         C(2,4,I) = C(2,4,I) - U24 * C(4,4,I)
  970        1                    - U23 * C(3,4,I)
  971         C(1,4,I) = C(1,4,I) - U14 * C(4,4,I)
  972        1                    - U13 * C(3,4,I) - U12 * C(2,4,I)
  973         C(3,5,I) = C(3,5,I) - U34 * C(4,5,I)
  974         C(2,5,I) = C(2,5,I) - U24 * C(4,5,I)
```

```
LINE #  SOURCE TEXT
 975        1                     - U23 * C(3,5,I)
 976         C(1,5,I) = C(1,5,I) - U14 * C(4,5,I)
 977        1                     - U13 * C(3,5,I) - U12 * C(2,5,I)
 978       3 CONTINUE
 979  C
 980         DO 6 I1        = 1 , 4
 981         DO 9 I         = 2 , IM1
 982       9 GG(I1,I,K)     = C(I1,1,I)
 983         DO 6 I2        = 1 , 4
 984         DO 6 I         = 2 , IM1
 985         EE(I1,I2,I,K) = C(I1,I2+1,I)
 986       6 CONTINUE
 987    1000 CONTINUE
 988  C
 989  C      BACKWARD SUBSTITUTION
 990  C
 991         DO 7 K      = KMAX - 3, 1 , - 1
 992         DO 7 I1     = 1 , 4
 993         DO 7 I      = 2 , IM1
 994         GG(I1,I,K) = GG(I1,I,K) - EE(I1,1,I,K) * GG(1,I,K+1)
 995        1                        - EE(I1,2,I,K) * GG(2,I,K+1)
 996        2                        - EE(I1,3,I,K) * GG(3,I,K+1)
 997        3                        - EE(I1,4,I,K) * GG(4,I,K+1)
 998       7 CONTINUE
 999         RETURN
1000         END
```

```
LINE #
1001        SUBROUTINE METRIC
1002  C*************************************************************************
1003  C*                                                                       *
1004  C*          SUBROUTINE METRIC                                            *
1005  C*                                                                       *
1006  C*************************************************************************
1007        PARAMETER (IX=180,KX=60)
1008        COMMON/FIX/OMEGA,HDOT
1009        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
1010        COMMON/GRID1/X(IX,KX),Z(IX,KX)
1011        COMMON/GRID/YACOB(IX,KX)
1012        COMMON/MTRIX/XIX(IX,KX),XIZ(IX,KX),ZETAX(IX,KX),ZETAZ(IX,KX),
1013       1XIT(IX,KX),ZETAT(IX,KX)
1014  C
1015  C***  THE SUBROUTINE METRIC COMPUTES THE METRICS IN ALL THE TWO DIRECTIONS AND
1016  C     THE UNSTEADY COEFFICIENTS ETAT ETC.
1017  C
1018        DO 1000 K = 1 , KMAX
1019        DO 1000 I = 1 , IMAX
1020        XTAU = OMEGA * Z(I,K)
1021        YTAU = OMEGA * (-X(I,K))- HDOT
1022  C***  PRESENT SET UP IS FOR FLOW PAST AN  AIRFOIL.
1023  C
1024        IF(I.EQ.1.OR.I.EQ.IMAX) GO TO 10
1025        XXI = .5 * (X(I+1,K)-X(I-1,K))
1026        ZXI = .5 * (Z(I+1,K)-Z(I-1,K))
1027        GO TO 15
1028     10 IF(I.EQ.IMAX) GO TO 16
1029        XXI = 1.0 * (X(2,K) - X(1,K))
1030        ZXI = 1.0 * (Z(2,K) - Z(1,K))
1031        GO TO 15
1032     16 XXI = 1.0 * (X(IMAX,K) - X(IMAX-1,K))
1033        ZXI = 1.0 * (Z(IMAX,K) - Z(IMAX-1,K))
1034     15 CONTINUE
1035        IF(K.EQ.1.OR.K.EQ.KMAX) GO TO 17
1036        XZET = .5 *(X(I,K+1)-X(I,K-1))
1037        ZZET = .5 *(Z(I,K+1)-Z(I,K-1))
1038        GO TO 20
1039     17 IF(K.EQ.KMAX) GO TO 18
1040        XZET = 2. * X(I,2)-1.5 * X(I,1) - .5 * X(I,3)
1041        ZZET = 2. * Z(I,2) - 1.5 * Z(I,1) - .5 * Z(I,3)
1042        GO TO 20
1043     18 XZET = 1.5 * X(I,KMAX)-2.* X(I,KMAX-1)+.5*X(I,KMAX-2)
1044        ZZET = 1.5 * Z(I,KMAX)-2.* Z(I,KMAX-1)+.5*Z(I,KMAX-2)
1045     20 CONTINUE
1046        YACOBI = XXI * ZZET - XZET * ZXI
1047        YACOB(I,K) = 1. / YACOBI
1048        XIX(I,K) = ZZET * YACOB(I,K)
1049        XIZ(I,K) = -XZET * YACOB(I,K)
1050        XTAU = OMEGA * Z(I,K)
1051        YTAU = - OMEGA * X(I,K)- HDOT
1052        XIT(I,K) = - XIX(I,K) * XTAU - XIZ(I,K) * YTAU
1053        ZETAX(I,K) = -ZXI * YACOB(I,K)
1054        ZETAZ(I,K) = XXI * YACOB(I,K)
1055        ZETAT(I,K) = - ZETAX(I,K) * XTAU - ZETAZ(I,K) * YTAU
1056  1000 CONTINUE
1057        RETURN
1058        END
```

```
      SUBROUTINE DISSIP
C*****************************************************************
C*                                                               *
C*          SUBROUTINE DISSIP                                    *
C*                                                               *
C*****************************************************************
      PARAMETER (IX=180,KX=60)
      COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
      COMMON/MTRIX/XIX(IX,KX),XIZ(IX,KX),ZETAX(IX,KX),ZETAZ(IX,KX),
     1XIT(IX,KX),ZETAT(IX,KX)
      COMMON/PERTR/DQ1(IX,KX),DQ2(IX,KX),DQ3(IX,KX),DQ4(IX,KX)
      COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
      COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
      COMMON/GRID/YACOB(IX,KX)
      COMMON/DAMP/WW,WW1,WW2X,WW2Y,WW4X,WW4Y
      COMMON/RADIUS/ SPECX(IX,KX),SPECY(IX,KX)
      COMMON/SPEED/ U(IX,KX),V(IX,KX),AA(IX,KX)
      DIMENSION P(IX),EPS(IX),DIS1(IX,4),DIS2(IX,4),DIS3(IX,4)
      DIMENSION QQ(4)
C
C     THIS SUBROUTINE ADDS THE FOURTH ORDER DISSIPATION TERMS TO THE
C     RIGHT HAND SIDE
C
C
      IM1 = IMAX - 1
      IM2 = IMAX - 2
      KM1 = KMAX - 1
      KM2 = KMAX - 2
      WDT = WW*DT
      GM1 = GAMMA - 1.
      DO 20 K = 2 , KM1
C
C     COMPUTE SWTICHING FUNCITON BASED ON SECOND DERIVAITVE OF PRESSURE
C
      DO 1 I = 1 , IMAX
    1 P(I)   = GM1 * (Q4(I,K) - 0.5*Q1(I,K)*(U(I,K)**2 + V(I,K)**2))
      DO 3 I = 2 , IM1
      PS1    = P(I+1) - 2.*P(I) + P(I-1)
      PS2    = P(I+1) + 2.*P(I) + P(I-1)
      EPS(I) = ABS(PS1/PS2)
    3 CONTINUE
      EPS(1)    = EPS(2)
      EPS(IMAX) = EPS(IM1)
C
C     SMOOTH OUT PRESSURE COEIFFICIENT
C
      DO 4 I = 2 , IM1
      P(I)  = AMAX1( EPS(I+1),EPS(I),EPS(I-1) )
    4 CONTINUE
      P(1)    = P(2)
      P(IMAX) = P(IM1)
C
      DO 10 I   = 2 , IM1
      C2        = WW2X * P(I)
      C4        = WW4X - AMIN1(WW4X,C2)
      C22       = C2*WDT * ( SPECX(I,K) + SPECX(I+1,K) )
      C44       = -C4*WDT* SPECX(I,K)
C     EPM        = Q4(I-1,K) + P(I-1)
C     EP         = Q4(I,K) + P(I)
C     EPP        = Q4(I+1,K) + P(I+1)
      DIS1(I,1) = C22* ( Q1(I+1,K) - Q1(I,K) )
      DIS1(I,2) = C22* ( Q2(I+1,K) - Q2(I,K) )
      DIS1(I,3) = C22* ( Q3(I+1,K) - Q3(I,K) )
      DIS1(I,4) = C22* ( Q4(I+1,K) - Q4(I,K) )
C     DIS1(I,4) = C22* ( EPP - EP )
      DIS2(I,1) = C44 * ( Q1(I+1,K) - 2.*Q1(I,K) + Q1(I-1,K) )
      DIS2(I,2) = C44 * ( Q2(I+1,K) - 2.*Q2(I,K) + Q2(I-1,K) )
      DIS2(I,3) = C44 * ( Q3(I+1,K) - 2.*Q3(I,K) + Q3(I-1,K) )
      DIS2(I,4) = C44 * ( Q4(I+1,K) - 2.*Q4(I,K) + Q4(I-1,K) )
C     DIS2(I,4) = C44 * ( EPP - 2.*EP + EPM )
   10 CONTINUE
C     B.C. TREATMENT
      QQ(1) = Q1(2,K) - Q1(1,K)
      QQ(2) = Q2(2,K) - Q2(1,K)
      QQ(3) = Q3(2,K) - Q3(1,K)
      QQ(4) = Q4(2,K) - Q4(1,K)
C     QQ(4) = Q4(2,K) + P(2) - Q4(1,K) - P(1)
      DO 15 N4       = 1 , 4
      C2             = WW2X*P(1)
      C22            = C2 * WDT * ( SPECX(1,K) + SPECX(2,K) )
      DIS1(1,N4)     = C22 * QQ(N4)
      DIS2(1,N4)     = 0.
   15 DIS2(IMAX,N4) = 0.
C
      DO 16 I   = 1 , IM1
      DIS3(I,1) = DIS1(I,1) + DIS2(I+1,1) - DIS2(I,1)
      DIS3(I,2) = DIS1(I,2) + DIS2(I+1,2) - DIS2(I,2)
      DIS3(I,3) = DIS1(I,3) + DIS2(I+1,3) - DIS2(I,3)
      DIS3(I,4) = DIS1(I,4) + DIS2(I+1,4) - DIS2(I,4)
   16 CONTINUE
C
C     FILL IN DISSIPATION TERMS
C
      DO 18 I  = 2 , IM1
      DQ1(I,K) = DIS3(I,1) - DIS3(I-1,1)
      DQ2(I,K) = DIS3(I,2) - DIS3(I-1,2)
      DQ3(I,K) = DIS3(I,3) - DIS3(I-1,3)
      DQ4(I,K) = DIS3(I,4) - DIS3(I-1,4)
   18 CONTINUE
   20 CONTINUE
C
C     K DIRECTION
C
      DO 100 I = 2 , IM1
C
C     COMPUTE SWTICHING FUNCTION BASED ON SECOND DERIVATIVE OF PRESSURE
C
      DO 31 K = 2 , KMAX
   31 P(K)    = GM1 * (Q4(I,K) - 0.5*Q1(I,K) * (U(I,K)**2 + V(I,K)**2))
      P(1) = ( 4.*P(2) - P(3) ) / 3.
      DO 33 K = 2 , KM1
      PS1    = P(K+1) - 2.*P(K) + P(K-1)
      PS2    = P(K+1) + 2.*P(K) + P(K-1)
      EPS(K) = ABS(PS1/PS2)
   33 CONTINUE
      EPS(1)    = EPS(2)
      EPS(KMAX) = EPS(KM1)
C
C     SMOOTH OUT PRESSURE COEFFICIENT
C
```

```
1179        DO 34 K = 2 , KM1
1180        P(K)   = AMAX1( EPS(K+1),EPS(K),EPS(K-1) )
1181     34 CONTINUE
1182        P(1)    = P(2)
1183        P(KMAX) = P(KM1)
1184  C
1185        DO 52 K   = 2 , KM1
1186        C2        = WW2Y * P(K)
1187        C4        = WW4Y - AMIN1(WW4Y,C2)
1188        C22       = C2 * WDT * ( SPECY(I,K) + SPECY(I,K+1) )
1189        C44       = -C4*WDT* SPECY(I,K)
1190  C     HPM       = Q4(I,K-1) + P(K-1)
1191  C     HP        = Q4(I,K) + P(K)
1192  C     HPP       = Q4(I,K+1) + P(K+1)
1193        DIS1(K,1) = C22 * ( Q1(I,K+1) - Q1(I,K) )
1194        DIS1(K,2) = C22 * ( Q2(I,K+1) - Q2(I,K) )
1195        DIS1(K,3) = C22 * ( Q3(I,K+1) - Q3(I,K) )
1196        DIS1(K,4) = C22 * ( Q4(I,K+1) - Q4(I,K) )
1197  C     DIS1(K,4) = C22 * ( HPP - HP )
1198        DIS2(K,1) = C44 * ( Q1(I,K+1) - 2.*Q1(I,K) + Q1(I,K-1) )
1199        DIS2(K,2) = C44 * ( Q2(I,K+1) - 2.*Q2(I,K) + Q2(I,K-1) )
1200        DIS2(K,3) = C44 * ( Q3(I,K+1) - 2.*Q3(I,K) + Q3(I,K-1) )
1201        DIS2(K,4) = C44 * ( Q4(I,K+1) - 2.*Q4(I,K) + Q4(I,K-1) )
1202  C     DIS2(K,4) = C44 * ( HPP - 2.*HP + HPM )
1203     52 CONTINUE
1204  C     B.C. TREATMENT
1205        QQ(1)   = Q1(I,2) - Q1(I,1)
1206        QQ(2)   = Q2(I,2) - Q2(I,1)
1207        QQ(3)   = Q3(I,2) - Q3(I,1)
1208        QQ(4)   = Q4(I,2) - Q4(I,1)
1209  C     QQ(4)   = Q4(I,2) + P(2) - Q4(I,1) - P(1)
1210        DO 59 N4       = 1 , 4
1211        C2             = WW2Y * P(1)
1212        C22            = C2 * WDT * ( SPECY(I,1) + SPECY(I,2) )
1213        DIS1(1,N4)     = C22 * QQ(N4)
1214        DIS2(1,N4)     = 0.
1215     59 DIS2(KMAX,N4) = 0.
1216  C
1217        DO 60 K   = 1 , KM1
1218        DIS3(K,1) = DIS1(K,1) + DIS2(K+1,1) - DIS2(K,1)
1219        DIS3(K,2) = DIS1(K,2) + DIS2(K+1,2) - DIS2(K,2)
1220        DIS3(K,3) = DIS1(K,3) + DIS2(K+1,3) - DIS2(K,3)
1221        DIS3(K,4) = DIS1(K,4) + DIS2(K+1,4) - DIS2(K,4)
1222     60 CONTINUE
1223  C
1224  C     FILL IN DISSIPATION TERMS
1225  C
1226        DO 65 K  = 2 , KM1
1227        DQ1(I,K) = DQ1(I,K) + DIS3(K,1) - DIS3(K-1,1)
1228        DQ2(I,K) = DQ2(I,K) + DIS3(K,2) - DIS3(K-1,2)
1229        DQ3(I,K) = DQ3(I,K) + DIS3(K,3) - DIS3(K-1,3)
1230        DQ4(I,K) = DQ4(I,K) + DIS3(K,4) - DIS3(K-1,4)
1231     65 CONTINUE
1232    100 CONTINUE
1233        RETURN
1234        END
```

```
LINE #
1235        SUBROUTINE SPECR(ISPEC)
1236  C*****************************************************************
1237  C*                                                               *
1238  C*              SUBROUTINE SPECR                                 *
1239  C*                                                               *
1240  C*****************************************************************
1241        PARAMETER (IX=180,KX=60)
1242        COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
1243        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
1244        COMMON/MTRIX/ XIX(IX,KX),XIZ(IX,KX),ZETAX(IX,KX),ZETAZ(IX,KX)
1245       1 ,XIT(IX,KX),ZETAT(IX,KX)
1246        COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
1247        COMMON/RADIUS/ SPECX(IX,KX),SPECY(IX,KX)
1248        COMMON/SPEED/ U(IX,KX),V(IX,KX),AA(IX,KX)
1249        COMMON/GRID/YACOB(IX,KX)
1250  C
1251  C     THIS SUBROUTINE COMPUTE THE SPECTRAL RADIUS FOR SCALING THE
1252  C     EXPLICIT AND IMPLICIT DISSIPATIONS
1253  C
1254        IM1 = IMAX - 1
1255        KM1 = KMAX - 1
1256        GAM = GAMMA * ( GAMMA - 1. )
1257        DO  5  K = 1 , KMAX
1258        DO  5  I = 1 , IMAX
1259        U(I,K) = Q2(I,K) / Q1(I,K)
1260        V(I,K) = Q3(I,K) / Q1(I,K)
1261        AA(I,K)= GAM * ( Q4(I,K)/Q1(I,K) - 0.5*(U(I,K)**2 + V(I,K)**2) )
1262        IF(AA(I,K).LT.0.) PRINT*,'NEGATIVE A*A = ',AA(I,K),' AT ',I,K
1263      5 CONTINUE
1264  C
1265  C
1266  C     COMPUTE IMPLICIT DISSIPATION SCALING
1267  C     SPECX = SPECTRAL RADIUS FOR XI-DIRECTION
1268  C     SPECY = SPECTRAL RADIUS FOR ZETA-DIRECTION
1269  C
1270        IF(ISPEC.EQ.1) THEN
1271          DO  20  K  =  1   ,  KMAX
1272          DO  20  I  =  1   ,  IMAX
1273          SPECX(I,K) = 1. / YACOB(I,K)
1274          SPECY(I,K) = 1. / YACOB(I,K)
1275     20   CONTINUE
1276        ELSEIF(ISPEC.EQ.2) THEN
1277          DO  30  K  =  1   ,  KMAX
1278          DO  30  I  =  1   ,  IMAX
1279          UCON      = U(I,K)*XIX(I,K) + V(I,K)*XIZ(I,K) + XIT(I,K)
1280          VCON      = U(I,K)*ZETAX(I,K) + V(I,K)*ZETAZ(I,K) + ZETAT(I,K)
1281          SPECX(I,K) = ABS(UCON) / YACOB(I,K)
1282          SPECY(I,K) = ABS(VCON) / YACOB(I,K)
1283     30   CONTINUE
1284        ELSEIF(ISPEC.EQ.3) THEN
1285          DO  40  K  =  1   ,  KMAX
1286          DO  40  I  =  1   ,  IMAX
1287          UCON       = U(I,K)*XIX(I,K) + V(I,K)*XIZ(I,K) + XIT(I,K)
1288          VCON       = U(I,K)*ZETAX(I,K) + V(I,K)*ZETAZ(I,K) + ZETAT(I,K)
1289          XI2        = XIX(I,K)**2 + XIZ(I,K)**2
1290          ZETA2      = ZETAX(I,K)**2 + ZETAZ(I,K)**2
1291          SPECX(I,K) = (ABS(UCON) + SQRT(AA(I,K)*XI2) )   / YACOB(I,K)
1292          SPECY(I,K) = (ABS(VCON) + SQRT(AA(I,K)*ZETA2) ) / YACOB(I,K)
1293     40   CONTINUE
1294        ENDIF
1295        RETURN
1296        END
```

```
1297        SUBROUTINE EXPBC(CL)
1298  C*************************************************************
1299  C*                                                           *
1300  C*              SUBROUTINE EXPBC                             *
1301  C*                                                           *
1302  C*************************************************************
1303        PARAMETER (IX=180,KX=60)
1304        COMMON/SURF/PSUR(IX)
1305        COMMON/GRID1/X(IX,KX),Z(IX,KX)
1306        COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF
1307        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
1308        COMMON/GRID/YACOB(IX,KX)
1309        COMMON/DAMP/WW,WW1,WW2X,WW2Y,WW4X,WW4Y
1310        COMMON/MTRIX/XIX(IX,KX),XIZ(IX,KX),ZETAX(IX,KX),ZETAZ(IX,KX),
1311       1XIT(IX,KX),ZETAT(IX,KX)
1312        COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
1313        COMMON/SPEED/ U(IX,KX),V(IX,KX),AA(IX,KX)
1314        COMMON/INITI/UINF,VINF
1315        COMMON/BCLOG/CIRCOR
1316        COMMON/LOGIC/RSTRT,PITCH,RAMP
1317        LOGICAL RSTRT,PITCH,RAMP
1318        LOGICAL CIRCOR
1319        DIMENSION C1(4)
1320        DIMENSION A(2,2),RHS(2)
1321        DATA PI/3.1415927/
1322        DATA CHORD/1./
1323  C
1324        GAMI  = 1. / GAMMA
1325        GAMM  = GAMMA - 1.
1326        GAMMI = 1. / GAMM
1327  C
1328  C     INVISCID AND VISCOUS B.C. ON SOLID WALL
1329  C
1330        DO 1 I=  ITEL , ITEU
1331        K = 3
1332        C1(1) = XIT(I,K)
1333        C1(2) = XIX(I,K)
1334        C1(3) = XIZ(I,K)
1335        UCON3 = (Q2(I,K)*C1(2)+Q3(I,K)*C1(3))
1336       1/Q1(I,K)
1337        K = 2
1338        C1(1) = XIT(I,K)
1339        C1(2) = XIX(I,K)
1340        C1(3) = XIZ(I,K)
1341        UCON2 = (Q2(I,K)*C1(2)+Q3(I,K)*C1(3))
1342       1/Q1(I,K)
1343        RHS(1) = 2. * UCON2 - UCON3 - XIT(I,1)
1344  C     FOR VISCOUS FLOWS SET UCON TO ZERO ALSO
1345        IF(REYREF.GT.0.) RHS(1) = - XIT(I,1)
1346        A(1,1) = XIX(I,1)
1347        A(1,2) = XIZ(I,1)
1348        A(2,1) = ZETAX(I,1)
1349        A(2,2) = ZETAZ(I,1)
1350        RHS(2) = - ZETAT(I,1)
1351        TEMP1 = A(1,1)
1352        TEMP2 = A(1,2)
1353        TEMP3 = A(2,1)
1354        TEMP4 = A(2,2)
1355        DEN = 1. /(TEMP1 * TEMP4 - TEMP2 * TEMP3)
1356        A(1,1) = A(2,2) * DEN
1357        A(1,2) = - TEMP2 * DEN
1358        A(2,1) = - TEMP3 * DEN
1359        A(2,2) = TEMP1 * DEN
1360        Q1(I,1) = 2. * Q1(I,2) - Q1(I,3)
1361        Q2(I,1) = Q1(I,1)*(A(1,1)*RHS(1)+A(1,2)*RHS(2))
1362        Q3(I,1) = Q1(I,1) *(A(2,1)*RHS(1)+A(2,2)*RHS(2))
1363      1 CONTINUE
1364        DO 10 I=ITEL ,ITEU
1365        P2=GAMM*(Q4(I,2)-0.5*Q1(I,2)*(U(I,2)**2+V(I,2)**2))
1366        P3=GAMM*(Q4(I,3)-0.5*Q1(I,3)*(U(I,3)**2+V(I,3)**2))
1367        P1=(4.*P2-P3)/3.
1368        PSUR(I)=(GAMMA*P1-1.)/(.7*AMINF**2)
1369     10 Q4(I,1)=P1/GAMM+0.5*Q1(I,1)*(U(I,1)**2+V(I,1)**2)
1370  C
1371  C     FAR FIELD BOUNDARY CONDITION ONLY  FOR STEADY FLOW
1372  C
1373        CIRC  = 0
1374        IF(PITCH.OR.RAMP) GO TO 999
1375        IF(AMINF.GT.1.) GO TO 65
1376  C
1377  C     CIRCULATION CORRECTION AT THE FAR FIELD IS BASED ON POTENTIAL
1378  C     VORTEX
1379  C
1380        BETA  = SQRT( 1. - AMINF**2 )
1381        IF(CIRCOR) CIRC = 0.25 * CHORD * CL * BETA * AMINF / PI
1382  C
1383        COSAL = COS(ALFA)
1384        SINAL = SIN(ALFA)
1385        AINF  = 1.
1386        HINF  = GAMMI + 0.5 * AMINF**2
1387  C
1388  C     CIRCQ = CIRCULATION CORRECTION QUANTITY
1389  C
1390        K = KMAX
1391        DO  60  I  =  2  ,  IMAX-1
1392        XLOC = X(I,K) - XREF
1393        ZLOC = Z(I,K)
1394        RADIUS = SQRT( XLOC**2 + ZLOC**2 )
1395        ANGLE  = ATAN2(ZLOC,XLOC)
1396        CIRCQ  = CIRC / ( RADIUS * ( 1. - (AMINF*SIN(ANGLE-ALFA))**2 ) )
1397        UF     = UINF + CIRCQ * SIN(ANGLE)
1398        VF     = VINF - CIRCQ * COS(ANGLE)
1399        AFSQ   = GAMM * ( HINF - 0.5*( UF**2 + VF**2 ) )
1400        AF     = SQRT(AFSQ)
1401  C
1402  C     NONREFLECTING B.C. BASE ON 1-D RIEMANN INVARIANTS
1403  C     ZETXN, ZETZN = NORMALIZED NORMAL VECTOR
1404  C
1405        ANOR = 1. / SQRT( ZETAX(I,K)**2 + ZETAZ(I,K)**2 )
1406        ZETXN = ZETAX(I,K) * ANOR
1407        ZETZN = ZETAZ(I,K) * ANOR
1408  C
1409  C     CHECK FOR INFLOW OR OUTFLOW
1410  C     FOR INFLOW:  R1, VELT, ENTROY ARE SPECIFIED AS FREE STREAM VALUES
1411  C                  R2 IS EXTRAPOLATED
1412  C     FOR OUTFLOW: R1 IS SPECIFIED AS FREE STREAM VALUE
1413  C                  R2, VELT, ENTROPY ARE EXTRAPOLATED
1414  C
1415        RHOEXT = Q1(I,K-1)
1416        RHOI   = 1.  / Q1(I,K-1)
```

```
LINE #                                                    SOURCE TEXT
1417        UEXT   = Q2(I,K-1) * RHOI
1418        VEXT   = Q3(I,K-1) * RHOI
1419        EEXT   = Q4(I,K-1)
1420        PEXT   = GAMM * ( EEXT - 0.5*RHOEXT*( UEXT**2 + VEXT**2 ) )
1421  C
1422  C     SET RIEMANN INVARIANTS R1, AND R2
1423  C     VELN = NORMAL VELOCITY, VELT = TANGENTIAL VELOCITY
1424  C
1425        R1 = ZETXN * UF + ZETZN * VF - 2. * AF * GAMMI
1426        R2 = ZETXN * UEXT + ZETZN * VEXT + 2. * SQRT( GAMMA * PEXT /
1427       1                                              RHOEXT ) * GAMMI
1428  C
1429        VELN = ( R1 + R2 ) * 0.5
1430        SPSQ = ( R2 - R1 ) * GAMM * 0.25
1431        A2   = SPSQ**2
1432  C
1433  C     SET OTHER FIXED OR EXTRAPOLATED VARIABLES
1434  C
1435        IF(VELN.LE.0.) THEN
1436          VELT  = ZETZN * UF - ZETXN * VF
1437          ENTRO = GAMMA
1438        ELSE
1439          VELT  = ZETZN * UEXT - ZETXN * VEXT
1440          ENTRO = RHOEXT**GAMMA / PEXT
1441        ENDIF
1442  C
1443  C     NOW COMPUTE FLOW VARIABLES
1444  C
1445        U(I,K) = ZETXN * VELN + ZETZN * VELT
1446        V(I,K) = ZETZN * VELN - ZETXN * VELT
1447        Q1(I,K) = ( A2 * ENTRO * GAMI ) ** GAMMI
1448        PRESS   = A2 * Q1(I,K) * GAMI
1449        Q2(I,K) = Q1(I,K) * U(I,K)
1450        Q3(I,K) = Q1(I,K) * V(I,K)
1451        Q4(I,K) = PRESS*GAMMI + 0.5 * Q1(I,K) * ( U(I,K)**2 + V(I,K)**2 )
1452     60 CONTINUE
1453        GO TO 67
1454     65 CONTINUE
1455  C
1456  C     B.C. FOR SUPERSONIC FLOW
1457  C
1458        K = KMAX
1459        DO  66  I  =  2  ,  IMAX-1
1460        RHOI  = 1. / Q1(I,K)
1461        U(I,K) = Q2(I,K) * RHOI
1462        V(I,K) = Q3(I,K) * RHOI
1463        ANOR   = 1. / SQRT( ZETAX(I,K)**2 + ZETAZ(I,K)**2 )
1464        ZETXN  = ZETAX(I,K) * ANOR
1465        ZETZN  = ZETAZ(I,K) * ANOR
1466        VELN   = ZETXN * U(I,K) + ZETZN * V(I,K)
1467        IF(VELN.GE.0.) THEN
1468          Q1(I,K) = Q1(I,K-1)
1469          Q2(I,K) = Q2(I,K-1)
1470          Q3(I,K) = Q3(I,K-1)
1471          Q4(I,K) = Q4(I,K-1)
1472        ENDIF
1473     66 CONTINUE
1474  C
1475     67 CONTINUE
1476  C
1477  C     OUTFLOW B.C. AT THE DOWNSTREAM OF C-GRID ONLY FOR INVISCID FLOW
1478  C
1479        IF(REYREF.LT.0.) THEN
1480        I = 1
1481        IP1 = 1
1482        SIGN = -1.
1483  C
1484  C     CHECK FOR SUPERSONIC FLOW
1485  C
1486        IF(AMINF.GT.1.) GO TO 75
1487  C
1488     72 CONTINUE
1489        DO  74  K  =  1  ,  KMAX
1490  C
1491  C     CORRECT FREE STREAM VALUES WITH CIRCULATION CORRECTION
1492  C
1493        XLOC = X(I,K) - XREF
1494        ZLOC = Z(I,K)
1495        RADIUS = SQRT( XLOC**2 + ZLOC**2 )
1496        ANGLE  = ATAN2(ZLOC,XLOC)
1497        CIRCQ  = CIRC / ( RADIUS * ( 1. - (AMINF*SIN(ANGLE-ALFA))**2 ) )
1498        UF     = UINF + CIRCQ * SIN(ANGLE)
1499        VF     = VINF - CIRCQ * COS(ANGLE)
1500        AFSQ   = GAMM * ( HINF - 0.5*( UF**2 + VF**2 ) )
1501        AF     = SQRT(AFSQ)
1502  C
1503  C     XIXH, XIZH = NORMALIZED HORIZONTAL VECTOR
1504  C
1505        ANOR   = 1. / SQRT( XIX(I,K)**2 + XIZ(I,K)**2 )
1506        XIXH   = XIX(I,K) * ANOR
1507        XIZH   = XIZ(I,K) * ANOR
1508  C
1509        RHOEXT = Q1(I+IP1,K)
1510        RHOI   = 1. / Q1(I+IP1,K)
1511        UEXT   = Q2(I+IP1,K)
1512        VEXT   = Q3(I+IP1,K)
1513        EEXT   = Q4(I+IP1,K)
1514        PEXT   = GAMM * ( EEXT - 0.5*RHOEXT*( UEXT**2 + VEXT**2 ) )
1515  C
1516  C     SET RIEMANN INVARIANTS R1, AND R2
1517  C
1518        R1 = XIXH * UF + XIZH * VF - SIGN * 2. * AF * GAMMI
1519        R2 = XIXH * UEXT + XIZH * VEXT + SIGN * 2. * SQRT( GAMMA * PEXT /
1520       1                                              RHOEXT ) * GAMMI
1521  C
1522        VELN = ( R1 + R2 ) * 0.5
1523        SPSQ = SIGN * ( R2 - R1 ) * GAMM * 0.25
1524        A2   = SPSQ**2
1525  C
1526  C     SET OTHER FIXED OR EXTRAPOLATED VARIABLES
1527  C
1528        IF(SIGN*VELN.LE.0. ) THEN
1529          VELT  = -XIZH * UF + XIXH * VF
1530          ENTRO = GAMMA
1531        ELSE
1532          VELT  = -XIZH * UEXT + XIXH * VEXT
1533          ENTRO = RHOEXT**GAMMA / PEXT
1534        ENDIF
1535  C
1536  C     COMPUTE FLOW VARIABLES
```

```
LINE #                                                    SOURCE TEXT
1537  C
1538        U(I,K) = XIXH * VELN - XIZH * VELT
1539        V(I,K) = XIZH * VELN + XIXH * VELT
1540        Q1(I,K) = ( A2 * ENTRO * GAMI ) ** GAMMI
1541        PRESS  = A2 * Q1(I,K) * GAMI
1542        Q2(I,K) = Q1(I,K) * U(I,K)
1543        Q3(I,K) = Q1(I,K) * V(I,K)
1544        Q4(I,K) = PRESS*GAMMI + 0.5 * Q1(I,K) * ( U(I,K)**2 + V(I,K)**2 )
1545     74 CONTINUE
1546        IF(I.EQ.IMAX) GO TO 79
1547        I = IMAX
1548        IP1 = -1
1549        SIGN = 1.
1550        GO TO 72
1551     79 CONTINUE
1552        GO TO 89
1553  C
1554  C     B.C. FOR SUPERSONIC FLOW
1555  C
1556     75 CONTINUE
1557        DO  84  K  =  1  ,  KMAX
1558  C
1559        RHOI  = 1. / Q1(I,K)
1560        U(I,K) = Q2(I,K) * RHOI
1561        V(I,K) = Q3(I,K) * RHOI
1562        ANOR   = 1. / SQRT( XIX(I,K)**2 + XIZ(I,K)**2 )
1563        XIXH   = XIX(I,K) * ANOR
1564        XIZH   = XIZ(I,K) * ANOR
1565        VELN   = SIGN * ( XIXH * U(I,K) + XIZH * V(I,K) )
1566  C
1567        IF(VELN.GE.0.) THEN
1568          Q1(I,K) = Q1(I+IP1,K)
1569          Q2(I,K) = Q2(I+IP1,K)
1570          Q3(I,K) = Q3(I+IP1,K)
1571          Q4(I,K) = Q4(I+IP1,K)
1572        ENDIF
1573     84 CONTINUE
1574        IF(I.EQ.IMAX) GO TO 89
1575        I = IMAX
1576        IP1 = -1.
1577        SIGN = 1.
1578        GO TO 75
1579     89 CONTINUE
1580  C
1581  C     DOWNSTREAM B.C. FOR VISCOUS FLOW
1582  C
1583        ELSEIF(REYREF.GT.0.) THEN
1584  C
1585  C     OUTFLOW B.C. FOR VISCOUS FLOW
1586  C     RHO, U AND V ARE SET TO THE VALUES OF THE NEXT INTERIOR POINTS
1587  C     PRESSURE IS EXTRAPOLATED AND THEN COMPUTE ENERGY (Q4)
1588  C
1589        DO  100  K  =  1  ,  KMAX
1590        I = 1
1591        Q1(I,K) = Q1(I+1,K)
1592        Q2(I,K) = Q2(I+1,K)
1593        Q3(I,K) = Q3(I+1,K)
1594        PEXT   = GAMM * ( Q4(I+1,K) - 0.5 * (Q2(I+1,K)**2 + Q3(I+1,K)**2)
1595       1                                   / Q1(I+1,K) )
1596        Q4(I,K) = PEXT/GAMM + 0.5 * ( Q2(I,K)**2 + Q3(I,K)**2 ) / Q1(I,K)
1597        I = IMAX
1598        Q1(I,K) = Q1(I-1,K)
1599        Q2(I,K) = Q2(I-1,K)
1600        Q3(I,K) = Q3(I-1,K)
1601        PEXT   = GAMM * ( Q4(I-1,K) - 0.5 * (Q2(I-1,K)**2 + Q3(I-1,K)**2)
1602       1                                   / Q1(I-1,K) )
1603        Q4(I,K) = PEXT/GAMM + 0.5 * ( Q2(I,K)**2 + Q3(I,K)**2 ) / Q1(I,K)
1604    100 CONTINUE
1605  C
1606        ENDIF
1607  C
1608  C     AVERAGE FLOW VARIABLES ACROSS WAKE CUT (FOR C-GRID)
1609  C
1610    999 CONTINUE
1611        DO  120  I  =  1  ,  ITEL - 1
1612        IU = IMAX + 1 - I
1613        Q1AVG = 0.5 * ( Q1(I,2) + Q1(IU,2) )
1614        Q2AVG = 0.5 * ( Q2(I,2) + Q2(IU,2) )
1615        Q3AVG = 0.5 * ( Q3(I,2) + Q3(IU,2) )
1616        Q1(I,1)  = Q1AVG
1617        Q1(IU,1) = Q1AVG
1618        Q2(I,1)  = Q2AVG
1619        Q2(IU,1) = Q2AVG
1620        Q3(I,1)  = Q3AVG
1621        Q3(IU,1) = Q3AVG
1622        PSLOW  = GAMM * ( Q4(I,2) - 0.5 * ( Q2(I,2)**2 + Q3(I,2)**2 ) /
1623       1                                    Q1(I,2) )
1624        PSUPP  = GAMM * ( Q4(IU,2) - 0.5 * ( Q2(IU,2)**2+Q3(IU,2)**2 ) /
1625       1                                    Q1(IU,2) )
1626        PSAVG  = 0.5 * ( PSLOW + PSUPP )
1627        Q4AVG  = PSAVG / GAMM + 0.5 * ( Q2(I,1)**2 + Q3(I,1)**2 ) /
1628       1                              Q1(I,1)
1629        Q4(I,1)  = Q4AVG
1630        Q4(IU,1) = Q4AVG
1631    120 CONTINUE
1632        RETURN
1633        END
```

```
LINE #                                                          SOURCE TEXT
1634        SUBROUTINE STRESS(ITN)
1635  C*****************************************************************
1636  C*                                                               *
1637  C*              SUBROUTINE STRESS                                *
1638  C*                                                               *
1639  C*****************************************************************
1640        PARAMETER (IX=190,KX=60)
1641        COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
1642        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
1643        COMMON/GRID1/X(IX,KX),Z(IX,KX)
1644        COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
1645        COMMON/SPEED/ U(IX,KX),V(IX,KX),AA(IX,KX)
1646        COMMON/PERTR/DQ1(IX,KX),DQ2(IX,KX),DQ3(IX,KX),DQ4(IX,KX)
1647        COMMON/MUTUR/CMU(IX,KX)
1648        DIMENSION RH1(IX),RH2(IX),RH3(IX),RH4(IX)
1649        COMMON/LOGIC/RSTRT,PITCH,RAMP
1650        LOGICAL RSTRT,PITCH,RAMP
1651  C     THIS SUBROUTINE ADDS VISCOUS TERMS TO THE RIGHT HAND SIDE
1652  C
1653  C     CMUL = LAMINAR OR MOLECULAR VISCOSITY
1654  C
1655        CMUL = AMI.  / REYREF
1656  c      CALL EDDY(CMUL)
1657        KMAXM1 = KMAX - 1
1658        IMAXM1 = IMAX - 1
1659        PR = 1.
1660  C
1661  C     COMPUTE TXX,TXY AND VISCOUS DISSIPATION AT I - 1 / 2
1662  C
1663        DO 30 K = 2 , KMAXM1
1664        DO 20 I = 2 , IMAX
1665        UXI = U(I,K) - U(I-1,K)
1666        VXI = V(I,K) - V(I-1,K)
1667        AXI = AA(I,K) - AA(I-1,K)
1668        UZET= .25*(U(I,K+1)-U(I,K    )+U(I-1,K+1)-U(I-1,K-1))
1669        VZET= .25*(V(I,K+1)-V(I,K-1)+V(I-1,K+1)-V(I-1,K-1))
1670        AZET= .25*(AA(I,K+1)-AA(I,K-1)+AA(I-1,K+1)-AA(I-1,K-1))
1671        XXI = X(I,K) - X(I-1,K)
1672        ZXI = Z(I,K) - Z(I-1,K)
1673        XZET= .25 * (X(I,K+1)-X(I,K-1)+X(I-1,K+1)-X(I-1,K-1))
1674        ZZET= .25 * (Z(I,K+1)-Z(I,K-1)+Z(I-1,K+1)-Z(I-1,K-1))
1675        YAC = XXI * ZZET - ZXI * XZET
1676        YAC = 1.  / YAC
1677        XIX = ZZET * YAC
1678        ZETAX= - ZXI * YAC
1679        XIZ = -XZET * YAC
1680        ZETAZ= XXI * YAC
1681        CNM = (  5 * (CMU(I,K) + CMU(I-1,K)) + CMUL ) * DT
1682        UX  = UXI * XIX + UZET * ZETAX
1683        VX  = VXI * XIX + VZET * ZETAX
1684        AX  = AXI * XIX + AZET * ZETAX
1685        UZ  = UXI * XIZ + UZET * ZETAZ
1686        VZ  = VXI * XIZ +   ET * ZETAZ
1687        AZ  = AXI * XIZ + AZET * ZETAZ
1688        TXX = -(-4. * UX + 2. * VZ) * CNM / 3.
1689        TXY = CNM * (UZ + VX)
1690        TYY = -CNM / 3. * (-4. * VZ + 2. * UX)
1691  C     R4  = ((U(I,K)+U(I-1,K))*TXX+(V(I,K)+V(I-1,K))*TXY)*0.5
1692  C    1      + CNM / PR/(GAMMA - 1.) * AX
1693  C     S4  = ((U(I,K)+U(I-1,K))*TXY+(V(I,K)+V(I-1,K))*TYY)*0.5
1694  C    1      + CNM / PR / (GAMMA - 1.) * AZ
1695  C     DEBUG
1696  C     TURN OFF ENRGY DISSIPATION AND DIFFUSION
1697        R4 = 0.
1698        S4 = 0.
1699        RH1(I) = 0.
1700        RH2(I) = (XIX * TXX + XIZ * TXY) / YAC
1701        RH3(I) = (XIX * TXY + XIZ * TYY) / YAC
1702     20 RH4(I) = (XIX * R4 + XIZ * S4) / YAC
1703        DO 30 I = 2 , IMAXM1
1704        DQ1(I,K) = DQ1(I,K) + RH1(I+1) - RH1(I)
1705        DQ2(I,K) = DQ2(I,K) + RH2(I+1) - RH2(I)
1706        DQ3(I,K) = DQ3(I,K) + RH3(I+1) - RH3(I)
1707        DQ4(I,K) = DQ4(I,K) + RH4(I+1) - RH4(I)
1708     30 CONTINUE
1709  C     IN THE Z DIRECTION
1710        DO 70 I = 2 , IMAXM1
1711        DO 60 K = 2 , KMAX
1712        UXI = .25 * (U(I+1,K)-U(I-1,K)+U(I+1,K-1)-U(I-1,K-1))
1713        VXI = .25 * (V(I+1,K)-V(I-1,K)+V(I+1,K-1)-V(I-1,K-1))
1714        AXI = .25 * (AA(I+1,K)-AA(I-1,K)+AA(I+1,K-1)-AA(I-1,K-1))
1715        XXI = .25 * (X(I+1,K)-X(I-1,K)+X(I+1,K-1)-X(I-1,K-1))
1716        ZXI = .25 * (Z(I+1,K)-Z(I-1,K)+Z(I+1,K-1)-Z(I-1,K-1))
1717        UZET = U(I,K) - U(I,K-1)
1718        VZET = V(I,K) - V(I,K-1)
1719        AZET = AA(I,K) - AA(I,K-1)
1720        XZET = X(I,K) - X(I,K-1)
1721        ZZET = Z(I,K) - Z(I,K-1)
1722        YAC = XXI * ZZET - ZXI * XZET
1723        YAC = 1. / YAC
1724        XIX = ZZET * YAC
1725        ZETAX= - ZXI * YAC
1726        XIZ = -XZET * YAC
1727        ZETAZ= XXI * YAC
1728        CNM = (  5 * (CMU(I,K) + CMU(I,K-1)) + CMUL) * DT
1729        UX  = UXI * XIX + UZET * ZETAX
1730        VX  = VXI * XIX + VZET * ZETAX
1731        AX  = AXI * XIX + AZET * ZETAX
1732        UZ  = UXI * XIZ + UZET * ZETAZ
1733        VZ  = VXI * XIZ + VZET * ZETAZ
1734        AZ  = AXI * XIZ + AZET * ZETAZ
1735        TXX = -(-4. * UX + 2. * VZ) * CNM / 3.
1736        TXY = CNM * (UZ + VX)
1737        TYY = -CNM / 3. * (-4. * VZ + 2. * UX)
1738  C     R4  = ((U(I,K)+U(I,K-1))*TXX+(V(I,K)+V(I,K-1))*TXY)*0.5
1739  C    1      + CNM / PR/(GAMMA - 1.) * AX
1740  C     S4  = ((U(I,K)+U(I,K-1))*TXY+(V(I,K)+V(I,K-1))*TYY)*0.5
1741  C    1      + CNM / PR / (GAMMA - 1.) * AZ
1742        R4 = 0
1743        S4 = 0
1744        RH1(K) = 0.
1745        RH2(K) = (ZETAX * TXX + ZETAZ * TXY) / YAC
1746        RH3(K) = (ZETAX * TXY + ZETAZ * TYY) / YAC
1747     60 RH4(K) = (ZETAX * R4 + ZETAZ * S4) / YAC
1748        DO 70 K = 2 , KMAXM1
1749        DQ1(I,K) = DQ1(I,K) + RH1(K+1) - RH1(K)
1750        DQ2(I,K) = DQ2(I,K) + RH2(K+1) - RH2(K)
1751        DQ3(I,K) = DQ3(I,K) + RH3(K+1) - RH3(K)
1752        DQ4(I,K) = DQ4(I,K) + RH4(K+1) - RH4(K)
1753     70 CONTINUE
```

```
LINE #  SOURCE TEXT
1754    C
1755          RETURN
1756          END
```

LINE # | SOURCE TEXT

```
1757        SUBROUTINE LOAD(CL,CDP,CDF,CM ALFAS,XREF)
1758  C*************************************************************
1759  C*                                                           *
1760  C*              SUBROUTINE LOAD                              *
1761  C*                                                           *
1762  C*************************************************************
1763        PARAMETER (IX=180,kx=60)
1764        COMMON/GRID1/X(IX,KX),Y(IX,KX)
1765        COMMON/SKINCF/CF(IX)
1766        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
1767        COMMON/SURF/PSUR(IX)
1768  C
1769  C     THIS SUBROUTINE COMPUTES THE INVISCID CONTRIBUTIONS
1770  C     TO LOADS ON THE AIRFOIL SURFACE
1771  C
1772        CL = 0.
1773        CD = 0.
1774        CDF = 0.
1775        CM = 0.
1776        DO 300 I = ITEL , ITEU - 1
1777        DX = X(I+1,1) - X(I,1)
1778    300 CDF = CDF + ( CF(I) + CF(I+1) ) * 0.5 * DX
1779        DO 400 I = ITEL , ITEU - 1
1780        XL = .5 * (X(I,1)+X(I+1,1))
1781        YL = .5 * (Y(I,1)+Y(I+1,1))
1782        DX = X(I+1,1) - X(I,1)
1783        DY = Y(I+1,1) - Y(I,1)
1784        CPA = PSUR(I+1) * .5 + PSUR(I) * .5
1785        DCL = CPA * (-DX)
1786        DCD = CPA * DY
1787        CL = CL + DCL
1788        CD = CD + DCD
1789    400 CM = CM + DCD * YL - DCL * ( XL - XREF )
1790  C
1791        DCL = CL * COS(ALFAS) - CD * SIN(ALFAS)
1792        CDP = CL * SIN(ALFAS) + CD * COS(ALFAS)
1793        CL = DCL
1794        RETURN
1795        END
```

```
LINE #  SOURCE TEXT
1796        SUBROUTINE WRAP(II,JJ,XSING,YSING,XP,YP,S0,A0,Y0)
1797   C*************************************************************
1798   C*                                                           *
1799   C*              SUBROUTINE WRAP                              *
1800   C*                                                           *
1801   C*************************************************************
1802        PARAMETER (IX=180,kx=60)
1803        DIMENSION S0(IX,4),Y0(60,4),A0(IX,4),XP(1),YP(1)
1804   C
1805   C    THIS SUBROUTINE UNWRAPS THE AIRFOIL AND STORES THE UNWRAPPED
1806   C    SURFACE IN ARRAYS A0 AND S0. IT ALSO DETERMINES THE STRETCHING
1807   C    IN THE ETA DIRECTION.
1808   C
1809        IMID = (I1 + 1) / 2
1810        DY   = .8 / (JJ - 2)
1811        DO 1 J = 2 , JJ
1812        Y = FLOAT(J-2) * DY
1813      1 Y0(J,1) = 1.25 * Y / (1. - Y * Y)
1814        Y0(1 , 1) = - Y0(3,1)
1815        PI = 4. * ATAN ( 1.)
1816        ANGL = PI + PI
1817        U = XP(1) - XSING
1818        V = YP(1) - YSING
1819        U = 1.
1820        V = 0.
1821        IIM1 = II - 1
1822        DO 2 I = 1 , II
1823        X11 = XP(I) - XSING
1824        Y11 = YP(I) - YSING
1825        ANGL = ANGL + ATAN2((U*Y11-V*X11),(U*X11+V*Y11))
1826        R    = SQRT(X11**2 + Y11**2)
1827        U    = X11
1828        V    =Y11
1829        R = SQRT(R)
1830        A0(I,1) = R * COS(.5 * ANGL)
1831      2 S0(I,1) = R * SIN(.5 * ANGL)
1832   C    WRITE (6,1000)
1833   C    WRITE (6,2000) (I,A0(I,1),S0(I,1),I = 1 , II)
1834        RETURN
1835     1000 FORMAT(1X,'UNWRAPPED COORDINATES IN THE TRANSFORMED PLANE')
1836     2000 FORMAT(I5 , 2F20.8)
1837        END
```

```
LINE #  SOURCE TEXT
1838        SUBROUTINE TABINT(XP,YP,XSING,YSING,N)
1839   C*************************************************************
1840   C*                                                           *
1841   C*              SUBROUTINE TABINT                            *
1842   C*                                                           *
1843   C*************************************************************
1844        PARAMETER (IX=180,kx=60)
1845        DIMENSION XP(IX),YP(IX),S0(IX),A0(IX)
1846        U = XP(1) - XSING
1847        V = YP(1) - YSING
1848        U = 1.
1849        V = 0.
1850        ANGL = 8. * ATAN(1.)
1851        DO 1 I = 1,N
1852        X11 = XP(I) - XSING
1853        Y11 = YP(I) - YSING
1854        ANGL = ANGL + ATAN2((U*Y11-V*X11),(U*X11+V*Y11))
1855        R = SQRT(X11**2 + Y11 ** 2)
1856        U = X11
1857        V = Y11
1858        R = SQRT(R)
1859        A0(I) = R * COS(ANGL * .5)
1860      1 S0(I) = R * SIN(ANGL * .5)
1861        DX =(A0(N)-A0(1))/96.
1862        A0ST = A0(1)
1863        DO 3 I = 1 , 97
1864        XX = FLOAT(I-1) * DX + A0ST
1865        CALL TAINT(A0,S0,XX,YY,N,3,NER,MON)
1866        XP(I) = XX * XX - YY * YY + XSING
1867      3 YP(I) = 2. * XX * YY + YSING
1868        RETURN
1869        END
```

```
LINE #  SOURCE TEXT
1870        SUBROUTINE TAINT(XTAB,FTAB,X,FX,N,K,NER,MON)
1871    C***********************************************************************
1872    C*                                                                     *
1873    C*              SUBROUTINE TAINT                                       *
1874    C*                                                                     *
1875    C***********************************************************************
1876        PARAMETER (IX=180,kx=50)
1877        DIMENSION XTAB(1),FTAB(1),T(10),C(10)
1878    C
1879    C     NASA - AMES SUBROUTINE FOR POLYNOMIAL INTERPOLATION
1880    C     OF A TABULATED FUNCTION
1881    C
1882        IF(N-K) 1 , 1 , 2
1883      1 NER = 2
1884        RETURN
1885      2 IF(K-9) 3,3,1
1886      3 IF(MON) 4,4,5
1887      5 IF(MON-2) 6,7,4
1888      4 J = 0
1889        NM1 = N - 1
1890        DO 8 I = 1 , NM1
1891        IF(XTAB(I) - XTAB(I+1)) 9,11,10
1892     11 NER = 3
1893        RETURN
1894      9 J = J-1
1895        GO TO 8
1896     10 J = J+1
1897      8 CONTINUE
1898        MON = 1
1899        IF(J) 12 , 6 , 6
1900     12 MON = 2
1901      7 DO 13 I = 1 , N
1902        IF(X - XTAB(I)) 14,14,13
1903     14 J = I
1904        GO TO 18
1905     13 CONTINUE
1906        GO TO 15
1907      6 DO 16 I = 1 , N
1908        IF(X-XTAB(I)) 16,17,17
1909     17 J = I
1910        GO TO 18
1911     16 CONTINUE
1912     15 J = N
1913     18 J = J - (K+1) / 2
1914        IF(J) 19,19,20
1915     19 J = 1
1916     20 M = J + K
1917        IF(M - N) 21,21,22
1918     22 J = J - 1
1919        GO TO 20
1920     21 KP1 = K + 1
1921        JSAVE = J
1922     26 DO 23 L = 1, KP1
1923        C(L) = X - XTAB(J)
1924        T(L) = FTAB(J)
1925     23 J = J+1
1926        DO 24 J = 1,K
1927        I = J+1
1928     25 T(I) = (C(J)*T(I)-C(I)*T(J))/(C(J)-C(I))
1929        I = I+1
1930        IF(I-KP1) 25,25,24
1931     24 CONTINUE
1932        FX = T(KP1)
1933        NER = 1
1934        RETURN
1935        END
```

```
LINE #  SOURCE TEXT
1936        SUBROUTINE SING(N2,N,X,Z,XLE,YLE,TEA,TES,XSING,YSING,CHD)
1937   C*************************************************************
1938   C*                                                           *
1939   C*              SUBROUTINE SING                              *
1940   C*                                                           *
1941   C*************************************************************
1942        PARAMETER (IX=180,kx=60)
1943   C
1944   C
1945   C    THIS SUBROUTINE COMPUTES SINGULAR POINT LOCATIONS.
1946   C
1947        DIMENSION X(2) , Z(2)
1948        NU = N2
1949        N1 = N2 + 1
1950        N3  = N2 - 1
1951        X1  = X(N1)
1952        Z1  = Z(N1)
1953        X2  = X(N2)
1954        Z2  = Z(N2)
1955        X3  = X(N3)
1956        Z3  = Z(N3)
1957        D1  = X2 ** 2 - X1 ** 2
1958        D2  = Z2 ** 2 - Z1 ** 2
1959        D3  = X2 - X1
1960        D4  = Z2 - Z1
1961        D5  = X3 ** 2 - X1 ** 2
1962        D6  = Z3 ** 2 - Z1 ** 2
1963        D7  = X3 - X1
1964        D8  = Z3 - Z1
1965        B   = (D7 * ( D1 + D2) - D3*(D5+D6))/(2.*(D7*D4-D3*D8))
1966        IF(ABS(D3).LT.ABS(D7)) GO TO 10
1967        A = (D1 + D2 - 2. * B * D4) / (2. * D3)
1968        GO TO 20
1969     10 A = (D5 + D6 - 2. * B * D8) / ( 2. * D7)
1970     20 CONTINUE
1971        R = SQRT((X2-A)** 2 + (Z2-B)**2)
1972        XLE = X(NU)
1973        YLE = Z(NU)
1974        CHD = X(1) - X(NU)
1975        A2 = (Z(2)-Z(1)) /(X(2) - X(1))
1976        A1 = (Z(N)-Z(N-1))/(X(N)-X(N-1))
1977        TES = .5 * (A1 + A2)
1978        TEA = A2 - A1
1979        TEA = TEA * 57.29578
1980        XSING = (A-XLE) /2.
1981        YSING = (B-YLE) / 2.
1982        RETURN
1983        END
```

```
LINE #
1984        SUBROUTINE AIRFOL(IT,IE,ILE)
1985  C*****************************************************************
1986  C*          SUBROUTINE AIRFOL                                    *
1987  C*****************************************************************
1988        PARAMETER (IX=180,KX=60)
1989        COMMON/GRID1/X(IX,KX),Z(IX,KX)
1990        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
1991        COMMON/YSYM/ISYM
1992        DIMENSION S0(IX,4),A0(IX,4),Y0(60,4),XP(IX),YP(IX),
1993       1E(IX),F(IX),B0(49)
1994        DIMENSION  XL(IX), XU(IX),  YL(IX), YU(IX),
1995       2                   XX(IX),  YY(IX)
1996  C
1997        DATA (B0(I),I=1,32)/1.,1.0414,1.0836,1.1270,1.1715,1.2175,1.2651,
1998       11.3145,1.3659,1.4199,1.4755,1.5349,1.5973,1.6636,1.7342,1.8099,
1999       21.8914,1.9799,2.0764,2.1829,2.3012,2.4341,2.5653,2.7597,2.9646,
2000       33.2106,3.5141,3.9019,4.4219,5.1687,6.3632,8.6809/
2001  C
2002        READ(5,*)  IBMAX,ISYM
2003        IF( ISYM .EQ. 0 ) THEN
2004        DO 101  I=1,IBMAX
2005        READ(5,*) XU(I), YU(I), YL(I)
2006    101 CONTINUE
2007        ELSE
2008        DO 102  I = 1,IBMAX
2009    102 READ(5,*) XU(I), YU(I)
2010        DO 103  I = 1,IBMAX
2011    103 YL(I) = - YU(I)
2012        ENDIF
2013  C
2014        DO 1000 I=1,IBMAX
2015        IU = I + IBMAX
2016        XX(IU) = XU(I)
2017        YY(IU) = YU(I)
2018        IL = IBMAX - I + 1
2019        XX(I)  = XU(IL)
2020        YY(I)  = YL(IL)
2021   1000 CONTINUE
2022  c
2023        IBMAX2 = 2*IBMAX
2024        DO 1010   I=1,IBMAX2
2025        XP(I) = XX(I)
2026        YP(I) = YY(I)
2027   1010 CONTINUE
2028        FNU = IBMAX
2029        FNL = IBMAX
2030  C
2031  C     THIS SUBROUTINE GENERATES SHEAR PARABOLIC C-GRID
2032  C     THE FOLLOWING SUBROUTINES ARE RELATED TO THE GRID GENERATION
2033  C         WRAP          SING
2034  C         TABINT        CLUSTR
2035  C         TAINT         STRTC
2036  C
2037        DO 8 I = 1 , 32
2038      8 A0(I,1) = B0(I)
2039        II = IMAX
2040        JJ =  KMAX-1
2041        IT =  31
2042        IE = 127
2043        IIP1 = II + 1
2044        IIM1 = II - 1
2045        IIJJ = II * JJ
2046        IIJJ2 = II * ( JJ-2)
2047        ILE  = (IT + IE ) / 2
2048        PI   = 4. * ATAN(1.)
2049        NU = FNU
2050        NL = FNL
2051        N  = NU + NL
2052        SCALE = 1./ ( XP(1) - XP(NL) )
2053        DO 3333 I=1,N
2054        XP(I) = XP(I) * SCALE
2055        YP(I) = YP(I) * SCALE
2056   3333 CONTINUE
2057        CALL SING(NU,N,XP,YP,XLE,ZLE,TEA,TES,XSING,YSING,CHD)
2058        CALL TABINT(XP,YP,XSING,YSING,N)
2059        NBODY = IE + 1 - IT
2060        DO 6791 I = 1 , NBODY
2061        L = I - 1
2062        E(IT+L) = XP(I)
2063    6791 F(IT+L) = YP(I)
2064        IEP1 = IE + 1
2065        SLOPT = TES * .75
2066        DO 438 I = IEP1 , II
2067        I1 = I +1 - IE
2068        E(I) = A0(I1,1)
2069        DXI = 1. / 48.
2070        D   = 4. / 3. * (E(I) - .25)
2071        F(I) = F(IE) + SLOPT * ALOG(D) / D
2072        L = IIP1 - I
2073        E(L) = E(I)
2074    438 F(L) = F(IT) + SLOPT * ALOG(D)/D
2075  C     WRITE (6,439)
2076  C439  FORMAT(2X,3H  I,19X,1HX,19X,1HY)
2077  C     WRITE (6,37) (I,E(I),F(I),I = 1 , II)
2078        CALL WRAP(II,JJ,XSING,YSING,E,F,S0,A0,Y0)
2079        DO 10 J = 2 , JJ
2080        DO 10 I = 1 , II
2081        X(I,J-1) = A0(I,1)**2 - (S0(I,1)+Y0(J,1))**2
2082     10 Z(I,J-1) = 2. * A0(I,1) * (S0(I,1)+Y0(J,1))
2083        RETURN
2084     37 FORMAT(I5,2F20.8)
2085        END
```

```
LINE #
2086        SUBROUTINE CLUSTR(DS)
2087  C*************************************************************
2088  C*                                                           *
2089  C*              SUBROUTINE CLUSTR                            *
2090  C*                                                           *
2091  C*************************************************************
2092        PARAMETER (IX=180,KX=60)
2093        COMMON/GRID1/X(IX,KX),Z(IX,KX)
2094        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
2095        DIMENSION S(60),XP(60),YP(60),R(60)
2096  C
2097  C      THIS SUBROUTINE CLUSTERS A GIVEN X,Z GRID SUCH THAT THE FIRST POINT IS AT
2098  C
2099        DO 100 I = 1 , IMAX
2100        S(1) = 0.
2101        XP(1) = X(I,1)
2102        YP(1) = Z(I,1)
2103        DO 10 K = 2 , KMAX
2104        XP(K) = X(I,K)
2105        YP(K) = Z(I,K)
2106     10 S(K) = SQRT((XP(K)-XP(K-1))**2+(YP(K)-YP(K-1))**2)
2107       1+S(K-1)
2108        SUMDX = S(KMAX)
2109        CALL STRTCH(SUMDX,DS,F1,KMAX,FACTOR)
2110  C     WRITE (6,200) I,FACTOR
2111        R(1) = 0.
2112        DR = DS
2113        DO 20 K = 2 , KMAX
2114        R(K) = R(K-1) + DR
2115        DR = DR * FACTOR
2116     20 CONTINUE
2117        RLAST = 1. / R(KMAX)
2118        DO 30 K = 2 , KMAX
2119        R1 = R(K) * RLAST * S(KMAX)
2120        CALL TAINT(S,XP,R1,XP1,KMAX,3,NER,MON)
2121        X(I,K) = XP1
2122        CALL TAINT(S,YP,R1,YP1,KMAX,3,NER,MON)
2123        Z(I,K) = YP1
2124     30 CONTINUE
2125    100 CONTINUE
2126  C     WRITE (6,115)
2127        DO 110 I = 1 , IMAX
2128        DX = X(I,2) - X(I,1)
2129        DY = Z(I,2) - Z(I,1)
2130        DN = SQRT(DX*DX+DY*DY)
2131  C     WRITE(6,120) I , DX , DY , DN
2132    110 CONTINUE
2133        RETURN
2134    115 FORMAT(5X,6HNORMAL,1X,8HDISTANCE,3H AT,4H THE,5H WALL,/
2135       1,5H     I,8X,2HDX,8X,2HDY,8X,2HDN,//)
2136    120 FORMAT(I5,3F10.5)
2137    200 FORMAT(I5,F10.5)
2138        END
```

LINE # | SOURCE TEXT

```
2139        SUBROUTINE STRTCH(SUMDX,DX1,F1,N1,R)
2140  C*************************************************************
2141  C*                                                           *
2142  C*              SUBROUTINE STRTCH                            *
2143  C*                                                           *
2144  C*************************************************************
2145        PARAMETER (IX=180,kx=60)
2146  C
2147  C     THIS SUBROUTINE DETERMINES A GEOMETRIC
2148  C     PROGRESSION OF GRID SPACING BETWEEN 1 AND N1 SUH THAT
2149  C     SUMSDX) EQUALS SUMDX. THE RATIO BETWEEN SUCCESSIVE
2150  C     SPACINGS IS R.
2151        N = N1 - 1
2152        R = 1.5
2153        E1 = 1.E-04
2154        E2 = 1.E-04
2155        DO 10 L = 1, 50
2156        F= (R-1) * SUMDX - DX1*(R**N-1)
2157        FP = SUMDX - DX1 * FLOAT(N) * R ** (N-1)
2158        RITER = R - F/ FP
2159  C     IF(1.E-02.LT.RITER.AND.RITER.LT.1.) RITER = 1.
2160  C     IF(1..LT.RITER.AND.RITER.LT.100.) RITER=.01
2161        IF(ABS(R-RITER).LT. R*E1) GO TO 1
2162        R = RITER
2163     10 CONTINUE
2164        R = 1.0001
2165  C     DX1 = DETOT/FLOAT(N1-1)
2166        RETURN
2167      1 R= RITER
2168        RETURN
2169        END
```

```
LINE #  SOURCE TEXT
2170        SUBROUTINE EDDY(CMUL)
2171  C*************************************************************
2172  C*                                                           *
2173  C*              SUBROUTINE EDDY                              *
2174  C*                                                           *
2175  C*************************************************************
2176        PARAMETER (IX=180,KX=60)
2177        COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
2178        COMMON/MUTUR/CMU(IX,KX)
2179        COMMON/SKINCF/CF(IX)
2180        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
2181        COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
2182        COMMON/GRID1/X(IX,KX),Z(IX,KX)
2183        COMMON/SPEED/ U(IX,KX),V(IX,KX),AA(IX,KX)
2184        DIMENSION TIN(KX),TOUT(KX),Y(KX),TRANS(IX),S(IX),UU(KX),UE(IX)
2185  C
2186        CMUPP = 1000. * CMUL
2187        KEDGE = KMAX
2188        ILE = ( ITEL + ITEU ) / 2
2189        CHORD = X(ITEU,1) - X(ILE,1)
2190        DO 100 I = 2 , IMAX - 1
2191  C     IF( ABS(X(I,1)).LT.( ABS(X(ILE,1)) + 0.05 * CHORD ) ) GO TO 100
2192        UDIF = 0.
2193        UMAX = 0.
2194        UMIN = 9999.
2195        FMAX = 0.1E-06
2196        YMAX = .1E-06
2197        FYMAX = 0.1E-06
2198        Y(1) = 0.
2199  C     COMPUTE TAU AT THE WALL
2200        UET = U(I,2) - U(I,1)
2201        VET = V(I,2) - V(I,1)
2202        XXI = X(I-1,1) - X(I-1,1)
2203        ZXI = Z(I-1,1) - Z(I-1,1)
2204        XET = 4. * X(I,2) - 3. * X(I,1) - X(I,3)
2205        ZET = 4. * Z(I,2) - 3. * Z(I,1) - Z(I,3)
2206        XXI = .5 * XXI
2207        ZXI = .5 * ZXI
2208        XET = .5 * XET
2209        ZET = .5 * ZET
2210        YAC = 1. / (XXI * ZET - ZXI * XET)
2211        OMEGA =  (UET * XXI + VET * ZXI ) * YAC
2212        TWALL = AMINF * OMEGA / REYREF
2213        CF(I) = 2. * TWALL / (AMINF**2)
2214        FACT = SQRT(Q1(I,1) * ABS(TWALL))*REYREF/(26.*AMINF)
2215        DO 10 K = 2 , KEDGE-1
2216        UXI = U(I-1,K) - U(I-1,K)
2217        VXI = V(I-1,K) - V(I-1,K)
2218        UET = U(I,K+1) - U(I,K-1)
2219        VET = V(I,K+1) - V(I,K-1)
2220        XXI = X(I-1,K) - X(I-1,K)
2221        ZXI = Z(I-1,K) - Z(I-1,K)
2222        XET = X(I,K+1) - X(I,K-1)
2223        ZET = Z(I,K+1) - Z(I,K-1)
2224        YAC = 1. / (XXI * ZET - ZXI * XET)
2225        OMEGA = ABS(UET*XXI-VET*ZXI-UXI*XET-VXI*ZET) * YAC
2226        UTOT  = SQRT(U(I,K)**2 + V(I,K)**2)
2227        UMAX  = AMAX1(UTOT,UMAX)
2228        UMIN  = AMIN1(UTOT,UMIN)
2229        Y(K) = SQRT((X(I,K)-X(I,K-1))**2+(Z(I,K)-Z(I,K-1))**2)+Y(K-1)
2230        F = Y(K) * OMEGA
2231        IF((Y(K)*FACT).GT.20.) GO TO 31
2232        IF(I.GT.ITEL.AND.I.LT.ITEU) F = F * (1. - EXP(-Y(K)*FACT))
2233     31 CONTINUE
2234  C
2235        IF(F.GT.FMAX) THEN
2236          FMAX = F
2237          YMAX = Y(K)
2238        ENDIF
2239        FCT = Y(K) * FACT
2240        IF(FCT.GT.20.) FCT = 20.
2241        FCT = ABS(FCT)
2242        EL = .4 * Y(K) * (1. - EXP(-FCT))
2243        TIN(K) = Q1(I,K) * EL * EL * OMEGA
2244        TIN(K) = ABS(TIN(K))
2245     10 CONTINUE
2246        UDIF = ABS(UMAX-UMIN)
2247        KSWTCH = 0
2248        FWAKE = YMAX * FMAX
2249        F1 = 0.25 * YMAX * UDIF **2 / FMAX
2250        IF(F1.LT.FWAKE) FWAKE = F1
2251        DO 20 K = 2 , KEDGE - 1
2252        FKLEB = 0.
2253        IF(ABS(Y(K)/YMAX).LT.1.E+04) THEN
2254        FKLEB = 1. / (1. + 5.5 * (0.3 * Y(K)/YMAX) ** 6)
2255        END IF
2256        TOUT(K) = .0168 * 1.6 * Q1(I,K) * FWAKE * FKLEB
2257        TOUT(K) = ABS(TOUT(K))
2258        IF(KSWTCH.NE.0) GO TO 20
2259        IF(TIN(K).GT.TOUT(K)) KSWTCH = K - 1
2260     20  CONTINUE
2261        DO 30 K = 2 , KEDGE - 1
2262        IF(K.LE.KSWTCH) THEN
2263        CMU(I,K) = ABS(TIN(K))
2264        ELSE
2265        CMU(I,K) = ABS(TOUT(K) )
2266        END IF
2267     30 CONTINUE
2268  C
2269  C     FREEZE THE VALUE OF EDDY VISCOSITY TO AN UPPER LIMIT
2270  C
2271  C     K = 1
2272  C 734 K = K + 1
2273  C     IF(K.GT.KEDGE) GO TO 736
2274  C 735   IF(CMU(I,K).LE.CMUPP) GO TO 734
2275  C       CMU(I,K) = CMUPP
2276  C       K = K + 1
2277  C       IF(K.LE.KEDGE) GO TO 735
2278  C 736 CONTINUE
2279     100 CONTINUE
2280        RETURN
2281        END
```

```
LINE #  SOURCE TEXT
2282        SUBROUTINE RESI
2283  C*************************************************************
2284  C*                                                           *
2285  C*              SUBROUTINE RESI                              *
2286  C*                                                           *
2287  C*************************************************************
2288        PARAMETER (IX=180,KX=60)
2289        COMMON/FIX/OMEGA,EDOT
2290        COMMON/PERTR/DQ1(IX,KX),DQ2(IX,KX),DQ3(IX,KX),DQ4(IX,KX)
2291        COMMON/GRID1/X(IX,KX),Z(IX,KX)
2292        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
2293        COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
2294        COMMON/SPEED/ U(IX,KX),V(IX,KX),AA(IX,KX)
2295        COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
2296        COMMON/COMRES/ RHS(IX,4)
2297        XTAU(I,K) = OMEGA * Z(I,K)
2298        YTAU(I,K) = - OMEGA * X(I,K)- EDOT
2299  C     THIS SUBROUTINE COMPUTES THE RESIDUAL ON THE RIGHT HAND
2300  C     SIDE ARISING FROM THE EULER- PART OF THE GOVERNING EQUATIONS
2301  C
2302  C    FLUX TERMS WITHIN THE XI- DERIVATIVE
2303        DO 100 K = 2 , KMAX - 1
2304        DO 10 I = 1 , IMAX
2305        UCON = U(I,K) * (Z(I,K+1)-Z(I,K-1))
2306       1 - V(I,K) * (X(I,K+1)-X(I,K-1))
2307        UCON = 0.25 * DT * UCON
2308        XIT = - XTAU(I,K) *(Z(I,K+1)-Z(I,K-1))
2309       1 + YTAU(I,K) * (X(I,K+1) - X(I,K-1))
2310        XIT = XIT * DT * 0.25
2311        UCON = UCON + XIT
2312        RHS(I,1) = UCON * Q1(I,K)
2313        P = (GAMMA-1.) * (Q4(I,K) - .5*Q1(I,K)*(U(I,K)**2 + V(I,K)**2) )
2314        RHS(I,2) = Q2(I,K) * UCON + P * DT * 0.25 * (Z(I,K+1) - Z(I,K-1))
2315        RHS(I,3) = Q3(I,K) * UCON - P * DT * 0.25 * (X(I,K+1)-X(I,K-1))
2316        RHS(I,4) = UCON * (Q4(I,K)+P) - XIT * P
2317     10 CONTINUE
2318        DO 11 I = 2 , IMAX - 1
2319        DQ1(I,K) = DQ1(I,K) - RHS(I+1,1) + RHS(I-1,1)
2320        DQ2(I,K) = DQ2(I,K) - RHS(I+1,2) + RHS(I-1,2)
2321        DQ3(I,K) = DQ3(I,K) - RHS(I+1,3) + RHS(I-1,3)
2322     11 DQ4(I,K) = DQ4(I,K) - RHS(I+1,4) + RHS(I-1,4)
2323    100 CONTINUE
2324  C
2325  C       FLUX TERMS WITHIN THE ETA- DERIVATIVE
2326  C
2327        DO 200 I = 2 , IMAX - 1
2328        DO 20 K = 1 , KMAX
2329        VCON = U(I,K) * (Z(I-1,K)-Z(I+1,K))
2330       1      +V(I,K) * (X(I+1,K)-X(I-1,K))
2331        VCON = VCON * 0.25 * DT
2332        ETAT = -XTAU(I,K) * (Z(I-1,K)-Z(I+1,K)) - YTAU(I,K)*
2333       1                                    (X(I+1,K)-X(I-1,K))
2334        ETAT = ETAT * 0.25 * DT
2335        VCON = VCON + ETAT
2336        RHS(K,1) = VCON * Q1(I,K)
2337        P = (GAMMA-1.) * (Q4(I,K) - 0.5 * Q1(I,K)*(U(I,K)**2 + V(I,K)**2))
2338        RHS(K,2) = VCON * Q2(I,K) +P * DT * .25 * (Z(I-1,K)-Z(I+1,K))
2339        RHS(K,3) = VCON * Q3(I,K) + P * DT * .25 * (X(I+1,K) - X(I-1,K))
2340        RHS(K,4) = VCON * (Q4(I,K)+P) - ETAT * P
2341     20 CONTINUE
2342        DO 21 K = 2 , KMAX - 1
2343        DQ1(I,K) = DQ1(I,K) - RHS(K+1,1) + RHS(K-1,1)
2344        DQ2(I,K) = DQ2(I,K) - RHS(K+1,2) + RHS(K-1,2)
2345        DQ3(I,K) = DQ3(I,K) - RHS(K+1,3) + RHS(K-1,3)
2346     21 DQ4(I,K) = DQ4(I,K) - RHS(K+1,4) + RHS(K-1,4)
2347    200 CONTINUE
2348  C 300 FORMAT(216,4E14.6)
2349        RETURN
2350        END
```

```
LINE #  SOURCE TEXT
2351        SUBROUTINE ROTGRID(IMAX,KMAX,DALFA)
2352   C*****************************************************************
2353   C*                                                               *
2354   C*              SUBROUTINE ROTGID                                *
2355   C*                                                               *
2356   C*****************************************************************
2357        PARAMETER (IX=180,kx=60)
2358   C    ROTATE GRID IN THE CLOCKWISE DIRECTION BY AN AMOUNT DALFA
2359        COMMON/GRID1/X(IX,KX),Z(IX,KX)
2360        CA = COS(DALFA)
2361        SA = - SIN(DALFA)
2362        DO 20 K = 1 , KMAX
2363        DO 20 I = 1 , IMAX
2364        X1 = X(I,K)
2365        Z1 = Z(I,K)
2366        X(I,K) = X1 * CA - Z1 * SA
2367     20 Z(I,K) = Z1 * CA + X1 * SA
2368        RETURN
2369        END
```

LINE # SOURCE TEXT

```
2370        SUBROUTINE CPPLOT(I1,I2,FMACH,X,Y)
2371  C*************************************************************
2372  C*                                                           *
2373  C*              SUBROUTINE CPPLOT                            *
2374  C*                                                           *
2375  C*************************************************************
2376        PARAMETER (IX=180,kx=60)
2377  C
2378  C     THIS SUBROUTINE PLOTS CP AT EQUAL INTERVALS IN THE MAPPED PLANE
2379  C
2380        COMMON/SURF/PSUR(IX)
2381        DIMENSION KODE(4),LINE(90),X(IX),Y(IX)
2382        DATA KODE/1H ,1H+,1HI,1H*/
2383        WRITE ( 6 , 2)
2384      2 FORMAT(50H0PLOT OF CP AT EQUAL INTERVALS IN THE MAPPED PLANE/
2385       1        10H0     X    ,10H      X/C  ,10H     CPL  ,10H      CPL   )
2386        CP0 = (1. + .2 * FMACH **2) ** 3.5 - 1.
2387        CP0 = CP0 / ( .7 * FMACH **2)
2388        K0 = 30. * CP0 + 4.5
2389        IMIN = (I2-I1)/2 + I1
2390        ILOW = 2 * IMIN
2391        CHD=X(I1) - X(IMIN)
2392        DO 12 I = 1 , 90
2393     12 LINE(I) = KODE(1)
2394        DO 34 I = IMIN  , I2
2395        K = 30. * (CP0 - PSUR(I)) + 4.5
2396        K1 = 30. * (CP0 -PSUR(ILOW-I)) + 4.5
2397        IF(K.LT.1) K = 1
2398        IF(K1.LT.1) K1 = 1
2399        IF(K.GT.90) K = 90
2400        IF(K1 .GT. 90) K1 = 90
2401        LINE(K0) = KODE(3)
2402        LINE(K) = KODE(2)
2403        LINE(K1) = KODE(4)
2404        XOC = (X(I) - X(IMIN)) / CHD
2405        WRITE (6,610) X(I),XOC,PSUR(ILOW-I),PSUR(I),LINE
2406        LINE(K1) = KODE(1)
2407     34 LINE(K) = KODE(1)
2408        RETURN
2409    600 FORMAT(4F10.4)
2410    610 FORMAT(4F10.4,90A1)
2411        END
```

LINE # | SOURCE TEXT

```
2412        SUBROUTINE OUTRVC(REREAL)
2413  C*************************************************************
2414  C*                                                           *
2415  C*              SUBROUTINE OUTRVC                            *
2416  C*                                                           *
2417  C*************************************************************
2418        PARAMETER (IX=180,KX=60)
2419        COMMON/PLOT/TITLE(10),NSTPT,RESD(3000),RES,CLE(3000),CDPE(3000)
2420        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
2421        COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
2422        COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
2423        COMMON/SURF/PSUR(IX)
2424        COMMON/SKINCF/CF(IX)
2425        COMMON/MUTUR/CMU(IX,KX)
2426        COMMON/TITL/ITITLE
2427        CHARACTER ITITLE*80
2428        DIMENSION FY(IX,KX)
2429        DUM = 0.
2430        CODE = 0.
2431        IRES = IFIX(RES)
2432        ALFAD = ALFA * 45. / ATAN(1.)
2433        CMUL = AMINF / REYREF
2434        DO 10 K = 1 , KMAX
2435        DO 10 I = 1 , IMAX
2436        FY(I,K) = 0.
2437     10 CMU(I,K) = CMU(I,K) / CMUL
2438        WRITE(4) ITITLE
2439        WRITE(4) IMAX,KMAX,ITEL,ITEU,AMINF,ALFAD,REREAL,DUM,NSTPT,GAMMA,
2440       + CODE,RES,DUM
2441        WRITE(4) Q1,Q2,Q3,Q4
2442        WRITE(4) RESD
2443        WRITE(4) PSUR
2444        WRITE(4) CF
2445        WRITE(4) CLE,CDPE
2446        WRITE(4) CMU
2447        WRITE(4) FY
2448        WRITE(4) TE
2449        WRITE(4) TE
2450        RETURN
2451        END
```

LINE # | SOURCE TEXT

```
2452        SUBROUTINE OUTNT(NOUT,NSTPP,TIME,REREAL)
2453        PARAMETER (IX=180,KX=60)
2454        COMMON/GRID1/X(IX,KX),Z(IX,KX)
2455        COMMON/PAR/GAMMA,REYREF,ALFA,ALFA1,REDFRE,AMINF,ALFAI
2456        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
2457        COMMON/FLOW/Q1(IX,KX),Q2(IX,KX),Q3(IX,KX),Q4(IX,KX)
2458        ITN = NSTPP
2459        PI  = 4.*ATAN(1.)
2460        ALFAD=ALFA*(180./PI)
2461              IF( ITN .EQ. 1*NOUT ) THEN
2462        REWIND 31
2463        WRITE (31) IMAX , KMAX
2464        WRITE (31) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2465        WRITE (31) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2466        WRITE (31) IMAX , KMAX
2467        WRITE (31) AMINF, ALFAD, REREAL, TIME
2468        WRITE (31) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2469        WRITE (31) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2470        WRITE (31) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2471        WRITE (31) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2472        REWIND 31
2473        END IF
2474              IF( ITN .EQ. 2*NOUT ) THEN
2475        REWIND 32
2476        WRITE (32) IMAX , KMAX
2477        WRITE (32) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2478        WRITE (32) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2479        WRITE (32) IMAX , KMAX
2480        WRITE (32) AMINF, ALFAD, REREAL, TIME
2481        WRITE (32) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2482        WRITE (32) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2483        WRITE (32) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2484        WRITE (32) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2485        REWIND 32
2486        END IF
2487              IF( ITN .EQ. 3*NOUT ) THEN
2488        REWIND 33
2489        WRITE (33) IMAX , KMAX
2490        WRITE (33) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2491        WRITE (33) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2492        WRITE (33) IMAX , KMAX
2493        WRITE (33) AMINF, ALFAD, REREAL, TIME
2494        WRITE (33) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2495        WRITE (33) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2496        WRITE (33) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2497        WRITE (33) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2498        REWIND 33
2499        END IF
2500              IF( ITN .EQ. 4*NOUT ) THEN
2501        REWIND 34
2502        WRITE (34) IMAX , KMAX
2503        WRITE (34) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2504        WRITE (34) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2505        WRITE (34) IMAX , KMAX
2506        WRITE (34) AMINF, ALFAD, REREAL, TIME
2507        WRITE (34) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2508        WRITE (34) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2509        WRITE (34) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2510        WRITE (34) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2511        REWIND 34
2512        END IF
2513              IF( ITN .EQ. 5*NOUT ) THEN
2514        REWIND 35
2515        WRITE (35) IMAX , KMAX
2516        WRITE (35) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2517        WRITE (35) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2518        WRITE (35) IMAX , KMAX
2519        WRITE (35) AMINF, ALFAD, REREAL, TIME
2520        WRITE (35) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2521        WRITE (35) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2522        WRITE (35) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2523        WRITE (35) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2524        REWIND 35
2525        END IF
2526              IF( ITN .EQ. 6*NOUT ) THEN
2527        REWIND 36
2528        WRITE (36) IMAX , KMAX
2529        WRITE (36) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2530        WRITE (36) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2531        WRITE (36) IMAX , KMAX
2532        WRITE (36) AMINF, ALFAD, REREAL, TIME
2533        WRITE (36) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2534        WRITE (36) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2535        WRITE (36) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2536        WRITE (36) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2537        REWIND 36
2538        END IF
2539              IF( ITN .EQ. 7*NOUT ) THEN
2540        REWIND 37
2541        WRITE (37) IMAX , KMAX
2542        WRITE (37) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2543        WRITE (37) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2544        WRITE (37) IMAX , KMAX
2545        WRITE (37) AMINF, ALFAD, REREAL, TIME
2546        WRITE (37) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2547        WRITE (37) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2548        WRITE (37) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2549        WRITE (37) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2550        REWIND 37
2551        END IF
2552              IF( ITN .EQ. 8*NOUT ) THEN
2553        REWIND 38
2554        WRITE (38) IMAX , KMAX
2555        WRITE (38) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2556        WRITE (38) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2557        WRITE (38) IMAX , KMAX
2558        WRITE (38) AMINF, ALFAD, REREAL, TIME
2559        WRITE (38) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2560        WRITE (38) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2561        WRITE (38) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2562        WRITE (38) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2563        REWIND 38
2564        END IF
2565              IF( ITN .EQ. 9*NOUT ) THEN
2566        REWIND 39
2567        WRITE (39) IMAX , KMAX
2568        WRITE (39) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2569        WRITE (39) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2570        WRITE (39) IMAX , KMAX
2571        WRITE (39) AMINF, ALFAD, REREAL, TIME
```

```
LINE #  SOURCE TEXT
2572        WRITE (39) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2573        WRITE (39) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2574        WRITE (39) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2575        WRITE (39) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2576        END IF
2577             IF( ITN .EQ. 10*NOUT ) THEN
2578        REWIND 40
2579        WRITE (40) IMAX , KMAX
2580        WRITE (40) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2581        WRITE (40) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2582        WRITE (40) IMAX , KMAX
2583        WRITE (40) AMINF, ALFAD, REREAL, TIME
2584        WRITE (40) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2585        WRITE (40) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2586        WRITE (40) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2587        WRITE (40) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2588        END IF
2589             IF( ITN .EQ. 11*NOUT ) THEN
2590        REWIND 41
2591        WRITE (41) IMAX , KMAX
2592        WRITE (41) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2593        WRITE (41) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2594        WRITE (41) IMAX , KMAX
2595        WRITE (41) AMINF, ALFAD, REREAL, TIME
2596        WRITE (41) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2597        WRITE (41) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2598        WRITE (41) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2599        WRITE (41) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2600        END IF
2601             IF( ITN .EQ. 12*NOUT ) THEN
2602        REWIND 42
2603        WRITE (42) IMAX , KMAX
2604        WRITE (42) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2605        WRITE (42) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2606        WRITE (42) IMAX , KMAX
2607        WRITE (42) AMINF, ALFAD, REREAL, TIME
2608        WRITE (42) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2609        WRITE (42) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2610        WRITE (42) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2611        WRITE (42) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2612        END IF
2613             IF( ITN .EQ. 13*NOUT ) THEN
2614        REWIND 43
2615        WRITE (43) IMAX , KMAX
2616        WRITE (43) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2617        WRITE (43) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2618        WRITE (43) IMAX , KMAX
2619        WRITE (43) AMINF, ALFAD, REREAL, TIME
2620        WRITE (43) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2621        WRITE (43) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2622        WRITE (43) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2623        WRITE (43) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2624        END IF
2625             IF( ITN .EQ. 14*NOUT ) THEN
2626        REWIND 44
2627        WRITE (44) IMAX , KMAX
2628        WRITE (44) ( ( X(I,K), I=1,IMAX ),  K=1,KMAX  )
2629        WRITE (44) ( ( Z(I,K), I=1,IMAX ),  K=1,KMAX  )
2630        WRITE (44) IMAX , KMAX
2631        WRITE (44) AMINF, ALFAD, REREAL, TIME
2632        WRITE (44) ( ( Q1(I,K), I=1,IMAX ), K=1,KMAX  )
2633        WRITE (44) ( ( Q2(I,K), I=1,IMAX ), K=1,KMAX  )
2634        WRITE (44) ( ( Q3(I,K), I=1,IMAX ), K=1,KMAX  )
2635        WRITE (44) ( ( Q4(I,K), I=1,IMAX ), K=1,KMAX  )
2636        END IF
2637        RETURN
2638        END
```

```
      PROGRAM AIRFGRID
      PARAMETER (IX=300,KX=100)
      COMMON/GRID1/X(IX,KX),Z(IX,KX)
      COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
      LOGICAL  VISCOUS
      READ(5,10)
      READ(5,100) IMAX,KMAX
      READ(5,10)
      READ(5,100) ITEL,ITEU
      READ(5,10)
      READ(5,200) VISCOUS
      READ(5,10)
      READ(5,300) DMIN
      READ(5,10)
      READ(5,*) AORAT , AOEXP , sdispl
      ILE = ( ITEU + ITEL ) / 2
      IUP = ( ITEU - ITEL ) / 2
      WRITE(6,1000) IMAX,KMAX,ITEL,ITEU,ILE,IUP,DMIN,AORAT
 1000 FORMAT('Imax = ',I10,10x,'kmax = ',I10,5x,/,
     1       'Trailing edge lower          =',I10,/,
     2       'Trailing edge upper          =',I10,/,
     2       'Leading edge                 =',I10,/,
     2       'Iupper - Ilower              =',I10,/,
     3       'Distanse of the first point =',f20.10,/,
     4       'Streching ratio              =',f20.10,/)
      CALL AIRFOL( AORAT , AOEXP , sdispl )
      IF( VISCOUS ) CALL CLUSTR( DMIN )
C
      WRITE(6,1100)
 1100 FORMAT(//,'GRID BOUNDARIES',/)
      RTEOUT = ABS( X(1,1) - X(ITEL,1) )
      RLEIN  = ABS( X(ILE,KMAX) - X(ILE,1) )
      IUP    = ILE + ( ITEU - ILE ) / 2
      RUP    = ABS( Z(IUP,KMAX) - Z(IUP,1) )
      WRITE(6,1200)  RTEOUT , RLEIN , RUP
 1200 FORMAT(5X,'Didtanse between trailing edge and outflow =',f20.10,/
     1       ,5x,'Distanse     "     leading  edge and inflow  =',f20.10,/
     2       ,5x,'Distanse of the body from the upper boudary=',f20.10,/)
      REWIND 21
      WRITE (21) IMAX,KMAX
      WRITE (21) (( X(I,K), I=1,IMAX), K=1,KMAX ),
     1           (( Z(I,K), I=1,IMAX), K=1,KMAX )
      STOP
   10 FORMAT(1X)
  100 FORMAT(2I5)
  200 FORMAT(3L5)
  300 FORMAT(4F10.0)
      END
```

```
LINE #
      SUBROUTINE METRIC
C*************************************************************************
C*                                                                       *
C*         SUBROUTINE METRIC                                             *
C*                                                                       *
C*************************************************************************
      PARAMETER (IX=300,KX=100)
      COMMON/FIX/OMEGA,HDOT
      COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
      COMMON/GRID1/X(IX,KX),Z(IX,KX)
      COMMON/GRID/YACOB(IX,KX)
      COMMON/MTRIX/XIX(IX,KX),XIZ(IX,KX),ZETAX(IX,KX),ZETAZ(IX,KX),
     1XIT(IX,KX),ZETAT(IX,KX)
C
C***  THE SUBROUTINE METRIC COMPUTES THE METRICS IN ALL THE TWO DIRECTIONS AND
C     THE UNSTEADY COEFFICIENTS ETAT ETC.
C
      DO 1000 K = 1 , KMAX
      DO 1000 I = 1 , IMAX
      XTAU = OMEGA * Z(I,K)
      YTAU = OMEGA * (-X(I,K))- HDOT
C***  PRESENT SET UP IS FOR FLOW PAST AN  AIRFOIL.
C
      IF(I.EQ.1.OR.I.EQ.IMAX) GO TO 10
      XXI = .5 * (X(I+1,K)-X(I-1,K))
      ZXI = .5 * (Z(I+1,K)-Z(I-1,K))
      GO TO 15
   10 IF(I.EQ.IMAX) GO TO 16
      XXI = 1.0 * (X(2,K) - X(1,K))
      ZXI = 1.0 * (Z(2,K) - Z(1,K))
      GO TO 15
   16 XXI = 1.0 * (X(IMAX,K) - X(IMAX-1,K))
      ZXI = 1.0 * (Z(IMAX,K) - Z(IMAX-1,K))
   15 CONTINUE
      IF(K.EQ.1.OR.K.EQ.KMAX) GO TO 17
      XZET = .5 *(X(I,K+1)-X(I,K-1))
      ZZET = .5 *(Z(I,K+1)-Z(I,K-1))
      GO TO 20
   17 IF(K.EQ.KMAX) GO TO 18
      XZET = 2. * X(I,2)-1.5 * X(I,1) - .5 * X(I,3)
      ZZET = 2. * Z(I,2) - 1.5 * Z(I,1) - .5 * Z(I,3)
      GO TO 20
   18 XZET = 1.5 * X(I,KMAX)-2.* X(I,KMAX-1)+.5*X(I,KMAX-2)
      ZZET = 1.5 * Z(I,KMAX)-2.* Z(I,KMAX-1)+.5*Z(I,KMAX-2)
   20 CONTINUE
      YACOBI = XXI * ZZET - XZET * ZXI
      YACOB(I,K) = 1. / YACOBI
      XIX(I,K) = ZZET * YACOB(I,K)
      XIZ(I,K) = -XZET * YACOB(I,K)
      XTAU = OMEGA * Z(I,K)
      YTAU = - OMEGA * X(I,K)- HDOT
      XIT(I,K) = - XIX(I,K) * XTAU - XIZ(I,K) * YTAU
      ZETAX(I,K) = -ZXI * YACOB(I,K)
      ZETAZ(I,K) = XXI * YACOB(I,K)
      ZETAT(I,K) = - ZETAX(I,K) * XTAU - ZETAZ(I,K) * YTAU
 1000 CONTINUE
      RETURN
      END
```

```
LINE #  SOURCE TEXT
  107       SUBROUTINE WRAP(II,JJ,XSING,YSING,XP,YP,S0,A0,Y0)
  108 C*************************************************************
  109 C*                                                           *
  110 C*              SUBROUTINE WRAP                              *
  111 C*                                                           *
  112 C*************************************************************
  113       PARAMETER (IX=300,KX=100)
  114       DIMENSION S0(IX,4),Y0(IX,4),A0(IX,4),XP(1),YP(1)
  115 C
  116 C     THIS SUBROUTINE UNWRAPS THE AIRFOIL AND STORES THE UNWRAPPED
  117 C     SURFACE IN ARRAYS A0 AND S0. IT ALSO DETERMINES THE STRETCHING
  118 C     IN THE ETA DIRECTION.
  119 C
  120       IMID = (II + 1) / 2
  121       DY   = .8 / (JJ - 2)
  122       DO 1 J = 2 , JJ
  123       Y = FLOAT(J-2) * DY
  124     1 Y0(J,1) = 1.25 * Y / (1. - Y * Y)
  125       Y0(1 , 1) = - Y0(3,1)
  126       PI = 4. * ATAN ( 1.)
  127       ANGL = PI + PI
  128       U = XP(1) - XSING
  129       V = YP(1) - YSING
  130       U = 1.
  131       V = 0
  132       IIM1 = II - 1
  133       DO 2 I = 1 , II
  134       X11 = XP(I) - XSING
  135       Y11 = YP(I) - YSING
  136       ANGL = ANGL + ATAN2((U*Y11-V*X11),(U*X11+V*Y11))
  137       R    = SQRT(X11**2 + Y11**2)
  138       U    = X11
  139       V    =Y11
  140       R = SQRT(R)
  141       A0(I,1) = R * COS(.5 * ANGL)
  142     2 S0(I,1) = R * SIN(.5 * ANGL)
  143 C     WRITE (6,1000)
  144 C     WRITE (6,2000) (I,A0(I,1),S0(I,1),I = 1 , II)
  145       RETURN
  146  1000 FORMAT(1X,'UNWRAPPED COORDINATES IN THE TRANSFORMED PLANE')
  147  2000 FORMAT(I5 , 2F20.8)
  148       END
```

LINE # | SOURCE TEXT

```
      SUBROUTINE TABINT(XP,YP,XSING,YSING,N,sdispl)
C*****************************************************************
C*                                                               *
C*            SUBROUTINE TABINT                                  *
C*                                                               *
C*****************************************************************
      PARAMETER (IX=300,KX=100)
      DIMENSION XP(IX),YP(IX),S0(IX),A0(IX),DTHXI(IX),DXW(IX),DX(IX)
      COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU,ILE
      PI = 4.*ATAN(1.)
      U = XP(1) - XSING
      V = YP(1) - YSING
      U = 1.
      V = 0.
      ANGL = 8. * ATAN(1.)
      DO 1 I = 1,N
      X11 = XP(I) - XSING
      Y11 = YP(I) - YSING
      ANGL = ANGL + ATAN2((U*Y11-V*X11),(U*X11+V*Y11))
      R = SQRT(X11**2 + Y11 ** 2)
      U = X11
      V = Y11
      R = SQRT(R)
      A0(I) = R * COS(ANGL * .5)
    1 S0(I) = R * SIN(ANGL * .5)
C
      A0L = A0(N) - A0(1)
      DXWS = 0.0
C
      A0ST = A0(1)
      idiv = 1 + ( iteu - itel )
C
      THDX  = 0.
      DTHXI(1) = 0.
      DTHXI(2) = pi* SDISPL/2.
      DTHX  = (1.- SDISPL ) * PI / (IDIV-1)
      DO 9   I= 3,IDIV
    9 DTHXI(I) = DTHXI(I-1) + DTHX
      DO 10                I = 1,IDIV
      THDX   =  (I-1)*DTHX
      if( i .eq. 2 .or. i .eq. idiv )  then
      if( i .eq. 2 )                          tbdx = .5*sdispl * pi + thdx
      if( i .eq. idiv   )                     thdx = ( 1.-sdispl)*pi
      else
      thdx = thdx
      endif
       THDX   =  DTHXI(I)
c       thdxd = (180/pi)*thdx
c      write(6,*) i, thdxd
      DXW(I) =  abs ( SIN( THDX ) )
   10 DXWS = DXWS + DXW(I)
      DO 20                I = 1,IDIV
   20 DX(I) = DXW(I) * A0L / DXWS
      DXI = 0.
      DO 3       I = 1 , IDIV
      DXI = DXI + DX(I)
      XX  = DXI   + A0ST
      CALL TAINT(A0,S0,XX,YY,N,3,NER,MON)
      XP(I) = XX * XX - YY * YY + XSING
    3 YP(I) = 2. * XX * YY + YSING
      RETURN
      END
```

LINE # | SOURCE TEXT

```
211  C
212        SUBROUTINE TAINT(XTAB,FTAB,X,FX,N,K,NER,MON)
213  C*************************************************************
214  C*                                                           *
215  C*              SUBROUTINE TAINT                             *
216  C*                                                           *
217  C*************************************************************
218        PARAMETER (IX=300,KX=100)
219        DIMENSION XTAB(1),FTAB(1),T(10),C(10)
220  C
221  C     NASA - AMES SUBROUTINE FOR POLYNOMIAL INTERPOLATION
222  C     OF A TABULATED FUNCTION
223  C
224        IF(N-K) 1 , 1 , 2
225      1 NER = 2
226        RETURN
227      2 IF(K-9) 3,3,1
228      3 IF(MON) 4,4,5
229      5 IF(MON-2) 6,7,4
230      4 J = 0
231        NM1 = N - 1
232        DO 8 I = 1 , NM1
233        IF(XTAB(I) - XTAB(I+1)) 9,11,10
234     11 NER = 3
235        RETURN
236      9 J = J-1
237        GO TO 8
238     10 J = J+1
239      8 CONTINUE
240        MON = 1
241        IF(J) 12 , 6 , 6
242     12 MON = 2
243      7 DO 13 I = 1 , N
244        IF(X - XTAB(I)) 14,14,13
245     14 J = I
246        GO TO 18
247     13 CONTINUE
248        GO TO 15
249      6 DO 16 I = 1 , N
250        IF(X-XTAB(I)) 16,17,17
251     17 J = I
252        GO TO 18
253     16 CONTINUE
254     15 J = N
255     18 J = J - (K-1) / 2
256        IF(J) 19,19,20
257     19 J = 1
258     20 M = J + K
259        IF(M - N) 21,21,22
260     22 J = J - 1
261        GO TO 20
262     21 KP1 = K + 1
263        JSAVE = J
264     26 DO 23 L = 1, KP1
265        C(L) = X - XTAB(J)
266        T(L) = FTAB(J)
267     23 J = J+1
268        DO 24 J = 1,K
269        I = J+1
270     25 T(I) = (C(J)*T(I)-C(I)*T(J))/(C(J)-C(I))
271        I = I+1
272        IF(I-KP1) 25,25,24
273     24 CONTINUE
274        FX = T(KP1)
275        NER = 1
276        RETURN
277        END
```

```
LINE #  SOURCE TEXT
   278      SUBROUTINE SING(N2,N,X,Z,XLE,YLE,TEA,TES,XSING,YSING,CHD)
   279  C*****************************************************************
   280  C*                                                               *
   281  C*             SUBROUTINE SING                                   *
   282  C*                                                               *
   283  C*****************************************************************
   284       PARAMETER (IX=300,KX=100)
   285  C
   286  C
   287  C    THIS SUBROUTINE COMPUTES SINGULAR POINT LOCATIONS.
   288  C
   289       DIMENSION X(2) , Z(2)
   290       NU = N2
   291       N1 = N2 + 1
   292       N3  = N2 - 1
   293       X1  = X(N1)
   294       Z1  = Z(N1)
   295       X2  = X(N2)
   296       Z2  = Z(N2)
   297       X3  = X(N3)
   298       Z3  = Z(N3)
   299       D1  = X2 ** 2 - X1 ** 2
   300       D2  = Z2 ** 2 - Z1 ** 2
   301       D3  = X2 - X1
   302       D4  = Z2 - Z1
   303       D5  = X3 ** 2 - X1 ** 2
   304       D6  = Z3 ** 2 - Z1 ** 2
   305       D7  = X3 - X1
   306       D8  = Z3 - Z1
   307       B   = (D7 * ( D1 + D2) - D3*(D5+D6))/(2.*(D7*D4-D3*D8))
   308       IF(ABS(D3).LT.ABS(D7)) GO TO 10
   309       A = (D1 + D2 - 2. * B * D4) / (2. * D3)
   310       GO TO 20
   311    10 A = (D5 + D6 - 2. * B * D8) / ( 2. * D7)
   312    20 CONTINUE
   313       R = SQRT((X2-A)** 2 + (Z2-B)**2)
   314       XLE = X(NU)
   315       YLE = Z(NU)
   316       CHD = X(1) - X(NU)
   317       A2 = (Z(2)-Z(1)) /(X(2) - X(1))
   318       A1 = (Z(N)-Z(N-1))/(X(N)-X(N-1))
   319       TES = .5 * (A1 + A2)
   320       TEA = A2 - A1
   321       TEA = TEA * 57.29578
   322       XSING = (A+XLE) /2.
   323       YSING = (B+YLE) / 2.
   324       RETURN
   325       END
```

LINE # | SOURCE TEXT

```
      SUBROUTINE AIRFOL( A0RAT , a0exp ,sdispl)
C*************************************************************
C*          SUBROUTINE AIRFOL                                *
C*************************************************************
      PARAMETER (IX=300,KX=100)
      COMMON/GRID1/X(IX,KX),Z(IX,KX)
      COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
      COMMON/YSYM/ISYM
      DIMENSION S0(IX,4),A0(IX,4),Y0(IX,4),XP(IX),YP(IX),
     1E(IX),F(IX),B0(IX)
      DIMENSION  XL(IX), XU(IX),  YL(IX), YU(IX),
     2                   XX(IX),  YY(IX)
C
c      DATA (B0(I),I=1,32)/1.,1.0414,1.0836,1.1270,1.1715,1.2175,1.2651,
c     11.3145,1.3659,1.4199,1.4755,1.5349,1.5973,1.6636,1.7342,1.8099,
c     21.8914,1.9799,2.0764,2.1829,2.3012,2.4341,2.5853,2.7597,2.9646,
c     33.2106,3.5141,3.9019,4.4219,5.1687,6.3632,8.6809/
C
      READ(5,*) ISYM, IBMAX
      IF( ISYM .EQ. 1 ) THEN
      DO 101  I=1,IBMAX
  101 READ(5,*) XU(I), YU(I)
      ELSE
      DO 102  I=1,IBMAX
      READ(5,*) XU(I), YU(I), YL(I)
  102 CONTINUE
      ENDIF
      IF( ISYM .EQ. 1 ) THEN
      DO 103  I=1,IBMAX
  103 YL(I) = - YU(I)
      ENDIF
C
      DO 1000 I=1,IBMAX
      IU = I + IBMAX
      XX(IU) = XU(I)
      YY(IU) = YU(I)
      IL = IBMAX - I + 1
      XX(I)  = XU(IL)
      YY(I)  = YL(IL)
 1000 CONTINUE
C
      IBMAX2 = 2*IBMAX
      DO 1010   I=1,IBMAX2
      XP(I) = XX(I)
      YP(I) = YY(I)
 1010 CONTINUE
      FNU = IBMAX
      FNL = IBMAX
C
C     THIS SUBROUTINE GENERATES SHEAR PARABOLIC C-GRID
C     THE FOLLOWING SUBROUTINES ARE RELATED TO THE GRID GENERATION
C         WRAP          SING
C         TABINT        CLUSTR
C         TAINT         STRTC
C
      A0(1,1)  = 1.
      DO 8 I = 2 , IBMAX
    8 A0(I,1) = ( A0(I-1,1) * A0RAT )**A0EXP
      PI   = 4. * ATAN(1.)
      NU =  FNU
      NL =  FNL
      N  = NU + NL
      IT = ITEL
      IE = ITEU
      ILE= ( ITEL + ITEU ) / 2
      II =  IMAX
      JJ =  KMAX-1
      IIP1 = II + 1
      IIM1 = II - 1
      IIJJ = II * JJ
      IIJJ2 = II * ( JJ-2)
C
      SCALE = 1./ ( XP(1) - XP(NL) )
      DO 3333 I=1,N
      XP(I) = XP(I) * SCALE
      YP(I) = YP(I) * SCALE
 3333 CONTINUE
      CALL SING(NU,N,XP,YP,XLE,ZLE,TEA,TES,XSING,YSING,CHD)
      CALL TABINT(XP,YP,XSING,YSING,N,sdispl)
      NBODY = IE + 1 - IT
      DO 6791 I = 1 , NBODY
      L = I - 1
      E(IT+L) = XP(I)
 6791 F(IT+L) = YP(I)
      IEP1 = IE + 1
      SLOPT = TES * .75
      DO 438 I = IEP1 , II
      I1 = I +1 - IE
      E(I) = A0(I1,1)
      DXI = 1. / 48.
      D   = 4. / 3. * (E(I) - .25)
      F(I) = F(IE) + SLOPT * ALOG(D) / D
      L = IIP1 - I
      E(L) = E(I)
  438 F(L) = F(IT) + SLOPT * ALOG(D)/D
C     WRITE (6,439)
C439  FORMAT(2X,3H  I,19X,1HX,19X,1HY)
C     WRITE (6,37) (I,E(I),F(I),I = 1 , II)
      CALL WRAP(II,JJ,XSING,YSING,E,F,S0,A0,Y0)
      DO 10 J = 2 , JJ
      DO 10 I = 1 , II
      X(I,J-1) = A0(I,1)**2 - (S0(I,1)+Y0(J,1))**2
   10 Z(I,J-1) = 2. * A0(I,1) * (S0(I,1)+Y0(J,1))
      RETURN
   37 FORMAT(I5,2F20.8)
      END
```

```
LINE #
 432        SUBROUTINE CLUSTR(DS)
 433  C*****************************************************************
 434  C*                                                               *
 435  C*              SUBROUTINE CLUSTR                                *
 436  C*                                                               *
 437  C*****************************************************************
 438        PARAMETER (IX=300,KX=100)
 439        COMMON/GRID1/X(IX,KX),Z(IX,KX)
 440        COMMON/DGRID/DT,IMAX,KMAX,ITEL,ITEU
 441        DIMENSION S(IX),XP(IX),YP(IX),R(IX)
 442  C
 443  C       THIS SUBROUTINE CLUSTERS A GIVEN X,Z GRID SUCH THAT THE FIRST POINT IS AT
 444  C
 445        DO 100 I = 1 , IMAX
 446        S(1) = 0.
 447        XP(1) = X(I,1)
 448        YP(1) = Z(I,1)
 449        DO 10 K = 2 , KMAX
 450        XP(K) = X(I,K)
 451        YP(K) = Z(I,K)
 452     10 S(K) = SQRT((XP(K)-XP(K-1))**2+(YP(K)-YP(K-1))**2)
 453       1+S(K-1)
 454        SUMDX = S(KMAX)
 455        CALL STRTCH(SUMDX,DS,F1,KMAX,FACTOR)
 456  C     WRITE (6,200) I,FACTOR
 457        R(1) = 0.
 458        DR = DS
 459        DO 20 K = 2 , KMAX
 460        R(K) = R(K-1) + DR
 461        DR = DR * FACTOR
 462     20 CONTINUE
 463        RLAST = 1. / R(KMAX)
 464        DO 30 K = 2 , KMAX
 465        R1 = R(K) * RLAST * S(KMAX)
 466        CALL TAINT(S,XP,R1,XP1,KMAX,3,NER,MON)
 467        X(I,K) = XP1
 468        CALL TAINT(S,YP,R1,YP1,KMAX,3,NER,MON)
 469        Z(I,K) = YP1
 470     30 CONTINUE
 471    100 CONTINUE
 472  C     WRITE (6,115)
 473        DO 110 I = 1 , IMAX
 474        DX = X(I,2) - X(I,1)
 475        DY = Z(I,2) - Z(I,1)
 476        DN = SQRT(DX*DX+DY*DY)
 477  C     WRITE(6,120) I , DX , DY , DN
 478    110 CONTINUE
 479        RETURN
 480    115 FORMAT(5X,6HNORMAL,1X,8HDISTANCE,3H AT,4H THE,5H WALL,/
 481       1,5H     I,8X,2HDX,8X,2HDY,8X,2HDN,//)
 482    120 FORMAT(I5,3F10.5)
 483    200 FORMAT(I5,F10.5)
 484        END
```

```
LINE #
485       SUBROUTINE STRTCH(SUMDX,DX1,F1,N1,R)
486     C*****************************************************************
487     C*                                                               *
488     C*              SUBROUTINE STRTCH                                *
489     C*                                                               *
490     C*****************************************************************
491       PARAMETER (IX=300,KX=100)
492     C
493     C     THIS SUBROUTINE DETERMINES A GEOMETRIC
494     C     PROGRESSION OF GRID SPACING BETWEEN 1 AND N1 SUM THAT
495     C     SUM=DX) EQUALS SUMDX. THE RATIO BETWEEN SUCCESSIVE
496     C     SPACINGS IS R.
497           N = N1 - 1
498           R = 1.5
499           E1 = 1.E-04
500           E2 = 1.E-04
501           DO 10 L = 1, 50
502           F= (R-1) * SUMDX - DX1*(R**N-1)
503           FP = SUMDX - DX1 * FLOAT(N) * R ** (N-1)
504           RITER = R - F/ FP
505     C     IF(1.E-02.LT.RITER.AND.RITER.LT.1.) RITER = 1.
506     C     IF(1..LT.RITER.AND.RITER.LT.100.) RITER=.01
507           IF(ABS(R-RITER).LT. R*E1) GO TO 1
508           R = RITER
509        10 CONTINUE
510           R = 1.0001
511     C     DX1 = DZTOT/FLOAT(N1-1)
512           RETURN
513         1 R= RITER
514           RETURN
515           END
```

LINE # | SOURCE TEXT

```
516       SUBROUTINE ROTGRID(IMAX,KMAX,DALFA)
517   C***********************************************************************
518   C*                                                                     *
519   C*              SUBROUTINE ROTGID                                      *
520   C*                                                                     *
521   C***********************************************************************
522       PARAMETER (IX=300,KX=100)
523   C     ROTATE GRID IN THE CLOCKWISE DIRECTION BY AN AMOUNT DALFA
524       COMMON/GRID1/X(IX,KX),Z(IX,KX)
525       CA = COS(DALFA)
526       SA = - SIN(DALFA)
527       DO 20 K = 1 , KMAX
528       DO 20 I = 1 , IMAX
529       X1 = X(I,K)
530       Z1 = Z(I,K)
531       X(I,K) = X1 * CA - Z1 * SA
532    20 Z(I,K) = Z1 * CA + X1 * SA
533       RETURN
534       END
```

20. Ms. L. Cowles 1
Naval Air Development Center
Street Road
Warminster, PA 18974-5000

21. Dr. W. Tseng 1
Naval Air Development Center
Street Road
Warminster, PA 18974-5000

22. Mr. D. Findlay 1
Naval Air Development Center
Street Road
Warminster, PA 18974-5000

23. Mr. M. Walters 1
Naval Air Development Center
Street Road
Warminster, PA 18974-5000

24. Mr. W. J. King 1
Technology Area Manager, Aircraft
Code 212
Office of Naval Technology
Arlington, VA 22217-5000

25. Dr. T. Cebeci 1
Professor and Chairman
Dept. of Aerospace Engineering
California State University
Long Beach, CA 90840

26. Capt. H. Helin 1
Program Manager, Directorate of Aerospace Sciences
AFOSRNA
Bolling AFB
Washington, D.C. 20332-6448

27. Dr. T. L. Doligalski 1
Chief, Fluid Mechanics Division
Army Research Office
P.O. Box 1221
Research Triangle Park, NC 27709-2211

29. Lt. Col. Steven Grohsmeyer 2
P.O. Box 732
Venice, FL 34284

10. Dr. R. M. Howard — 1
Dept. of Aeronautics and Astronautics, Code 67HO
Naval Postgraduate School
Monterey, CA 93943

11. Dr. S. K. Hebbar — 1
Dept. of Aeronautics and Astronautics, Code 67HB
Naval Postgraduate School
Monterey, CA 93943

12. Dr. M. S. Chandrasekhara — 1
NASA Ames Research Center (M.S. 260-1)
Moffet Field, CA 94035

13. Dr. L. W. Carr — 1
NASA Ames Research Center (M.S. 260-1)
Moffet Field, CA 94035

14. Dr. L. B. Schiff — 1
NASA Ames Research Center (M.S. 258-1)
Moffet Field, CA 94035

15. Mr. D. P. Bencze — 1
Chief, Applied Aerodynamics Branch
NASA Ames Research Center (M.S. 227-6)
Moffet Field, CA 94035

16. Dr. S. Davis — 1
NASA Ames Research Center (M.S. 260-1)
Moffet Field, CA 94035

17. Mr. T. S. Momiyama — 1
Director, Aircraft Division
Code AIR-931
Naval Air Systems Command
Washington, D.C. 20631-9320

18. Mr. B. Neuman — 1
Aircraft Division
Code AIR-931
Naval Air Systems Command
Washington, D.C. 20631-9320

19. Mr. G. Derderian — 1
Aircraft Division
Code AIR-931
Naval Air Systems Command
Washington, D.C. 20631-9320

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Commandant of the Marine Corps Code TE 06 Headquarters, U.S. Marine Corps Washington, D.C. 20380-0001	1
2.	Library, Code 52 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Dr. J. A. Ekaterinaris NASA Ames Research Center (M.S. 260-1) Moffet Field, CA 94035	5
4.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
5.	Dr. E. R. Wood Chairman Dept. of Aeronautics and Astronautics, Code 67WD Naval Postgraduate School Monterey, CA 93943-5002	1
6.	Dr. M. F. Platzer Dept. of Aeronautics and Astronautics, Code 67PL Naval Postgraduate School Monterey, CA 93943-5002	5
7.	Dr. L. V. Schmidt Dept. of Aeronautics and Astronautics, Code 67SC Naval Postgraduate School Monterey, CA 93943-5002	1
8.	Dr. R. P. Shreeve Dept. of Aeronautics and Astronautics, Code 67SF Naval Postgraduate School Monterey, CA 93943-5002	1
9.	Dr. R. Kolar Dept. of Aeronautics and Astronautics, Code 67KJ Naval Postgraduate School Monterey, CA 93943-5002	1